



Sentinel from Scratch

Agenda

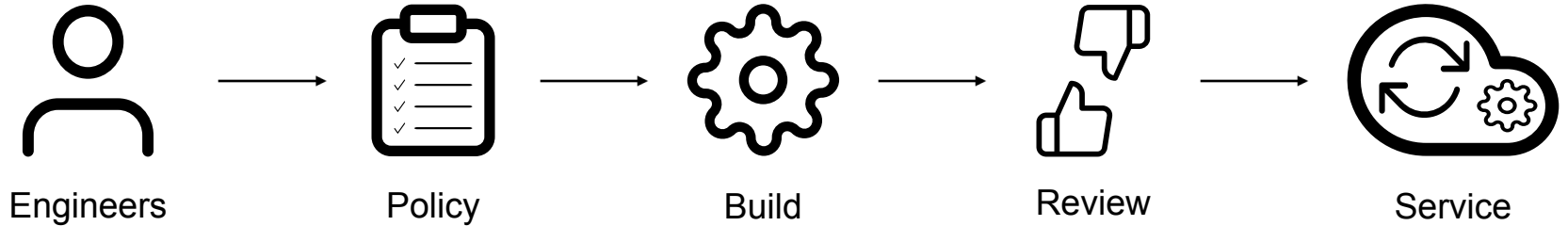


- Policy-as-Code
- Sentinel CLI
- Policies
- Questions



Policy-as-Code

Policy-as-Code



Policy-as-Code



- Treat policies as applications
- Store in version control
- Automate enforcement and review
- Automate logic testing
- Proactive vs. reactive



Sentinel CLI



Sentinel CLI

Available commands

```

$ sentinel

Usage: sentinel [--version] [--help] <command> [<args>]

Available commands are:
  apply      Execute a policy and output the result
  fmt        Format Sentinel policy to a canonical format
  test       Test policies
  version    Prints the Sentinel runtime version

```



Sentinel CLI

Contextual help

TERMINAL

```
$ sentinel --help apply
```

```
Usage: sentinel apply [options] POLICY
```

```
Execute the policy file specified by POLICY.
```

```
This runs the policy and outputs whether the policy passed or failed.  
The exit code also reflects the status of the policy: 0 = pass, 1 = fail,  
2 = undefined (fail, but result was undefined), 3 = error.
```

```
A configuration file is typically specified with -config to define the  
available imports, mock data, and global values. This is used to simulate  
a policy embedded within a host system.
```

```
Options:
```

```
-color          Enable or disable colorized output. Enabled by  
                default if running interactively.
```

```
-config FILE    Set the configuration file. Default:  
                sentinel.[hcl|json].
```

```
-global 'VAR=VALUE' Set a global value.  
                  This flag can be set multiple times.
```

```
....
```




Sentinel CLI

Evaluate policies locally

TERMINAL

```
$ sentinel apply -trace policy.sentinel
```

```
Sentinel Result: true
```

```
This result means that Sentinel policies returned true and the protected  
behavior is allowed by Sentinel policies.
```

```
1 policies evaluated.
```

```
## Policy 1: policy.sentinel (hard-mandatory)
```

```
Result: true
```

```
Description:
```

```
Exercise 01 - Main
```



Policies



Policies

File and folder structure

```
$ tree
.
├── modules
│   └── library.sentinel
├── policy.sentinel
├── sentinel.hcl
├── test
│   └── policy
│       ├── fail.hcl
│       └── pass.hcl
└── testdata
    └── http.sentinel
```

TERMINAL



Policies

Main

```
CODE EDITOR
```

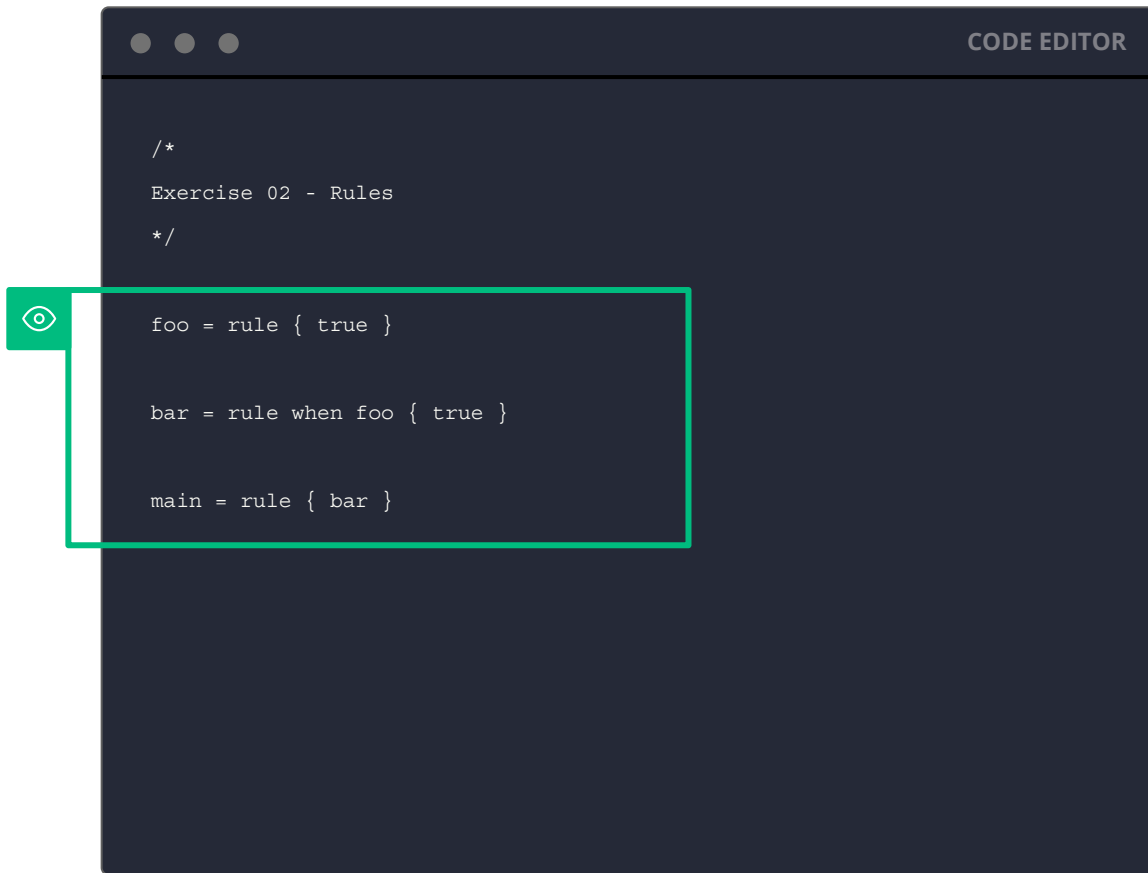
```
/*  
Exercise 01 - Main  
*/
```

```
main = true
```



Policies

Rules

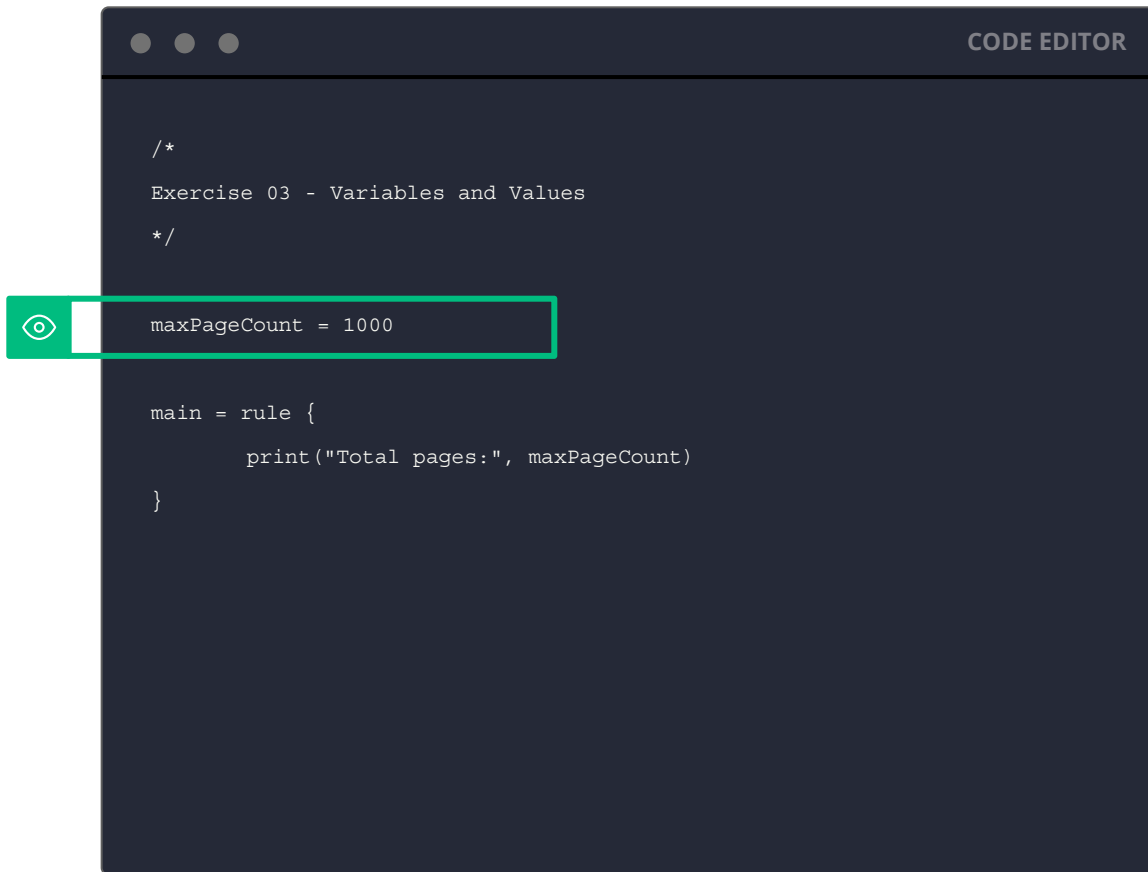


```
/*  
Exercise 02 - Rules  
*/  
  
foo = rule { true }  
  
bar = rule when foo { true }  
  
main = rule { bar }
```



Policies

Variables and Values



```
/*  
Exercise 03 - Variables and Values  
*/  
  
maxPageCount = 1000  
  
main = rule {  
    print("Total pages:", maxPageCount)  
}
```



Policies

Imports and Parameters

CODE EDITOR

```
/*  
Exercise 04 - Imports and Parameters  
*/  
  
import "types"  
  
param maxPageCount default 1000  
  
main = rule {  
    print("Total pages:", maxPageCount) and  
    print("pageCount is of type:", types.type_of(maxPageCount))  
}
```



Policies

Operators and Delimiters



CODE EDITOR

```
/*  
Exercise 05 - Operators & Delimiters  
*/  
  
import "types"  
  
param maxPageCount default 1000  
  
eval_bookPageCount = rule {  
    maxPageCount <= 1500  
}  
  
main = rule {  
    eval_bookPageCount  
}
```




Policies

Collections

```
CODE EDITOR

/*
Exercise 06 - Collections
*/

books = [
  {
    "title": "Unlocking Android",
    "pageCount": 416,
    ...
  },
  {..}
]

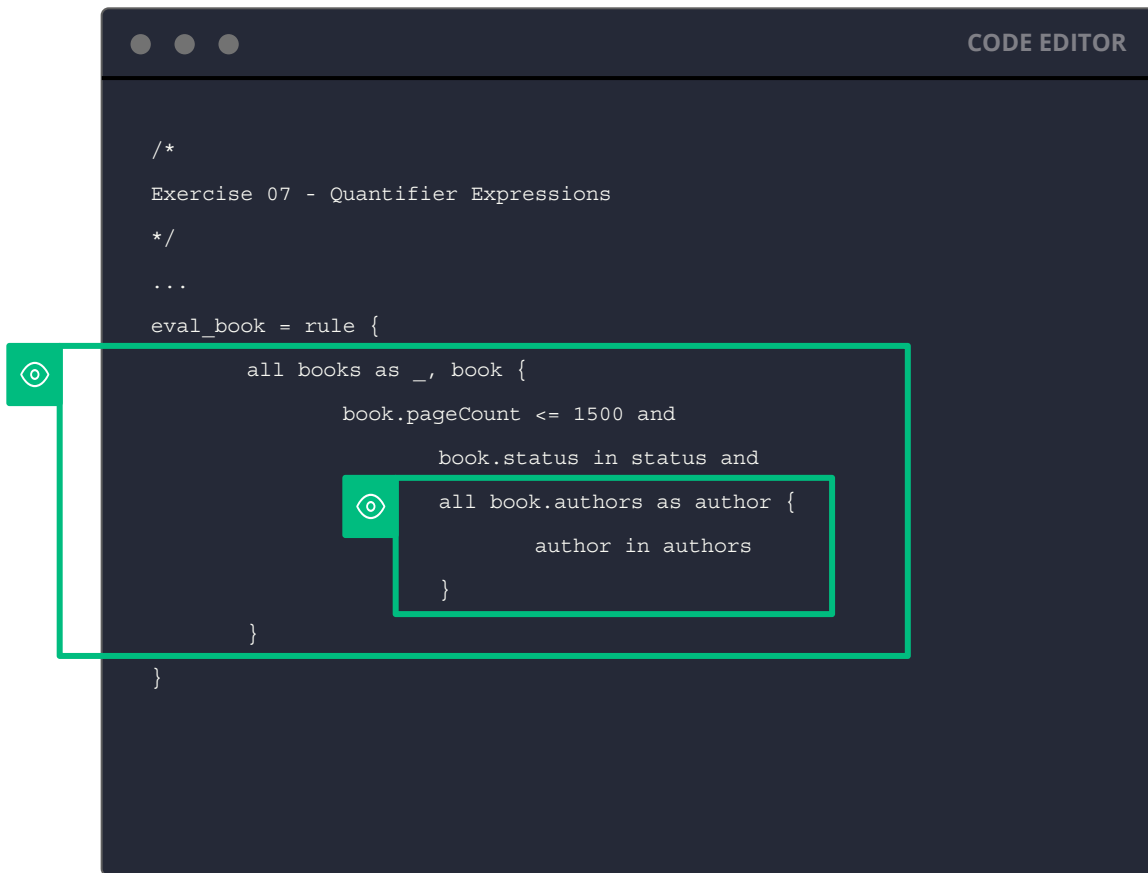
status = [
  "PUBLISH",
  "PENDING",
]

...
```



Policies

Quantifier Expressions



```
/*  
Exercise 07 - Quantifier Expressions  
*/  
...  
eval_book = rule {  
    all books as _, book {  
        book.pageCount <= 1500 and  
        book.status in status and  
        all book.authors as author {  
            author in authors  
        }  
    }  
}
```



Policies

Functions and Modules

```
CODE EDITOR

/*
Exercise 08 - Functions & Modules
*/

...

import "library"

...

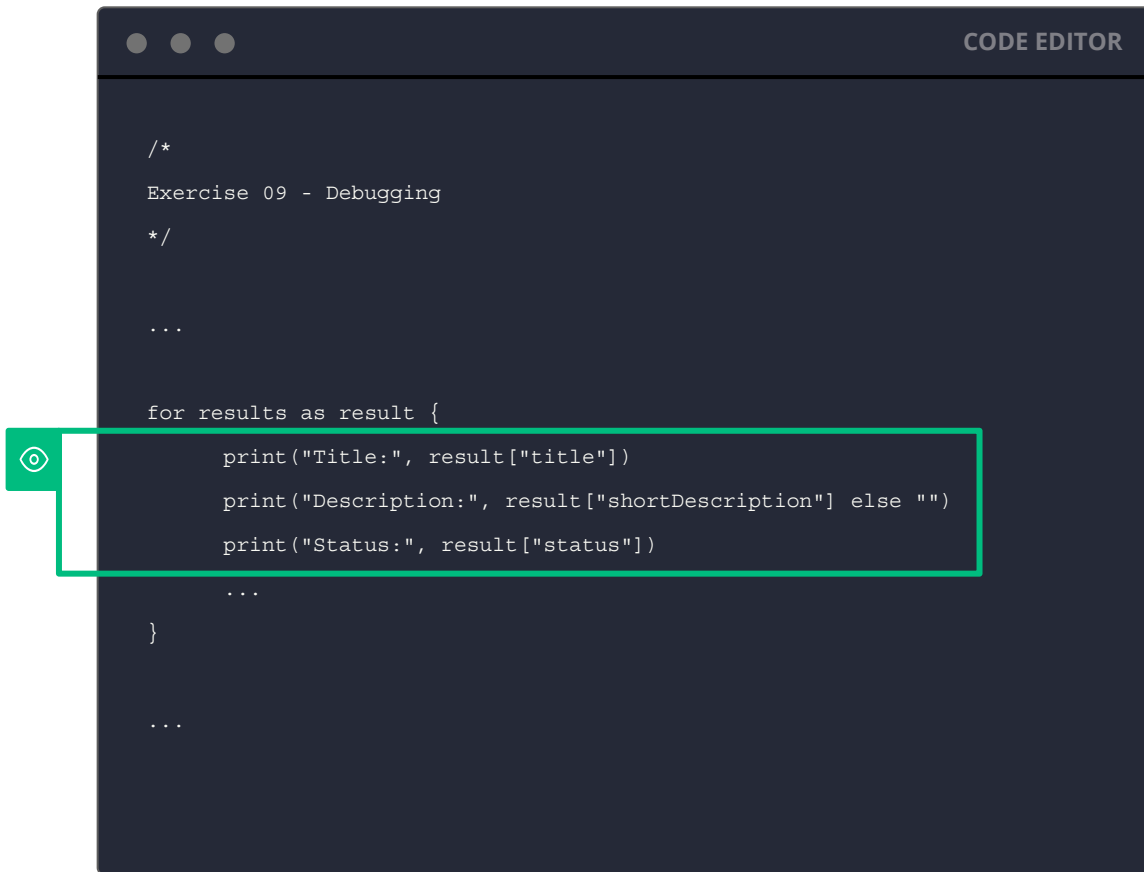
books = filter library.search.all() as _, book {
  book.status in status and
    all book.authors as author {
      author in authors
    }
}

...
```



Policies

Debugging



```
/*
Exercise 09 - Debugging
*/

...

for results as result {
    print("Title:", result["title"])
    print("Description:", result["shortDescription"] else "")
    print("Status:", result["status"])
    ...
}

...
```



Policies

Triage: tfplan.plan

 This plan was saved to: tfplan.plan

```

$ terraform plan -out=tfplan.plan

Refreshing Terraform state in-memory prior to plan...
The refreshed state will be used to calculate this plan, but will not be
persisted to local or remote state storage.

...

Plan: 5 to add, 0 to change, 0 to destroy.

-----

To perform exactly these actions, run the following command to apply:
  terraform apply "tfplan.plan"
```



Policies

Triage: plan.json



```

$ terraform show -json tfplan.plan > plan.json

$ cat plan.json | jq
{
  "format_version": "0.1",
  "terraform_version": "0.13.5",
  "planned_values": {...},
  "resource_changes": [...],
  "configuration": {...}
}
```



Policies

Triage: schemas.json



```

$ terraform providers schema -json > schemas.json

$ cat schemas.json | jq
{
  "format_version": "0.1",
  "provider_schemas": {
    "registry.terraform.io/hashicorp/null": {
      "provider": {...},
      "resource_schemas": {...},
      "data_source_schemas": {...}
    }
  }
}
```



Policies

Triage: mock generation



TERMINAL

```
$ tfe-sentinel-mock-generator
Generating data for current-version imports (Terraform 0.12 and higher).

Wrote import mock type "tfconfig" to <PATH>/mock-tfconfig.sentinel
Wrote import mock type "tfconfig_v2" to <PATH>/mock-tfconfig-v2.sentinel
Wrote import mock type "tfplan" to <PATH>/mock-tfplan.sentinel
Wrote import mock type "tfplan_v2" to <PATH>/mock-tfplan-v2.sentinel
Wrote import mock type "tfstate" to <PATH>/mock-tfstate.sentinel
Wrote import mock type "tfstate_v2" to <PATH>/mock-tfstate-v2.sentinel
Wrote import mock type "sentinel_config" to <PATH>/sentinel.json

Done. 7 files written.
```




Questions?



Docs and links

[Download Sentinel](#)

[Sentinel Language](#)

[Sentinel Language Specification](#)

[Sentinel Playground](#)

[Sentinel TFE Mock Generator](#)