Scribble Generator

Annotate your existing property tests A specification language like Scribble enables property writing, but it's not the only way to do property-based testing. Some exiting smart contract fuzzing tools use an alternative method. Instead of annotating functions, variables and contracts, you would write solidity functions that "encode" a property.

```

Copy // An example of a function which encodes an invariant over variable x function variable_x_is_positive() returns (bool) { return x > 0; }

```

How each tool works is slightly different. Some tools use the return value of functions, while others use a global variable to signal property failure to a fuzzer. These differences make it challenging to switch between tools and techniques.

Scribble Generator targets this problem. It adds annotations to all of your existing fuzz test cases. These annotations instantly enable any Scribble compatible tool to check the properties.

```

Copy

// This function can be universified as follows: //#if_succeeds result; function variable_x_is_positive() returns (bool) { return x > 0; }

```

Installation

Scribble Generator is not part of the base scribble package and has to be installed separately.

```

Copy npm install -g scribble-generator

```

Usage

Using Scribble Generator is straightforward. First move to a directory which contains solidity files (files in subdirectories are also included). Then execute the following command:

```

Copy scribble-generate

```

This will add Scribble annotations for all fuzz tests which weren't annotated yet.

Supported Tools

Currently we support automatic generation of annotations for testcases written for the following tools:

- Echidna
- Dapptools
-

Previous Introduction Next Installation Last updated2 years ago

On this page * Installation * Usage * Supported Tools

Was this helpful?