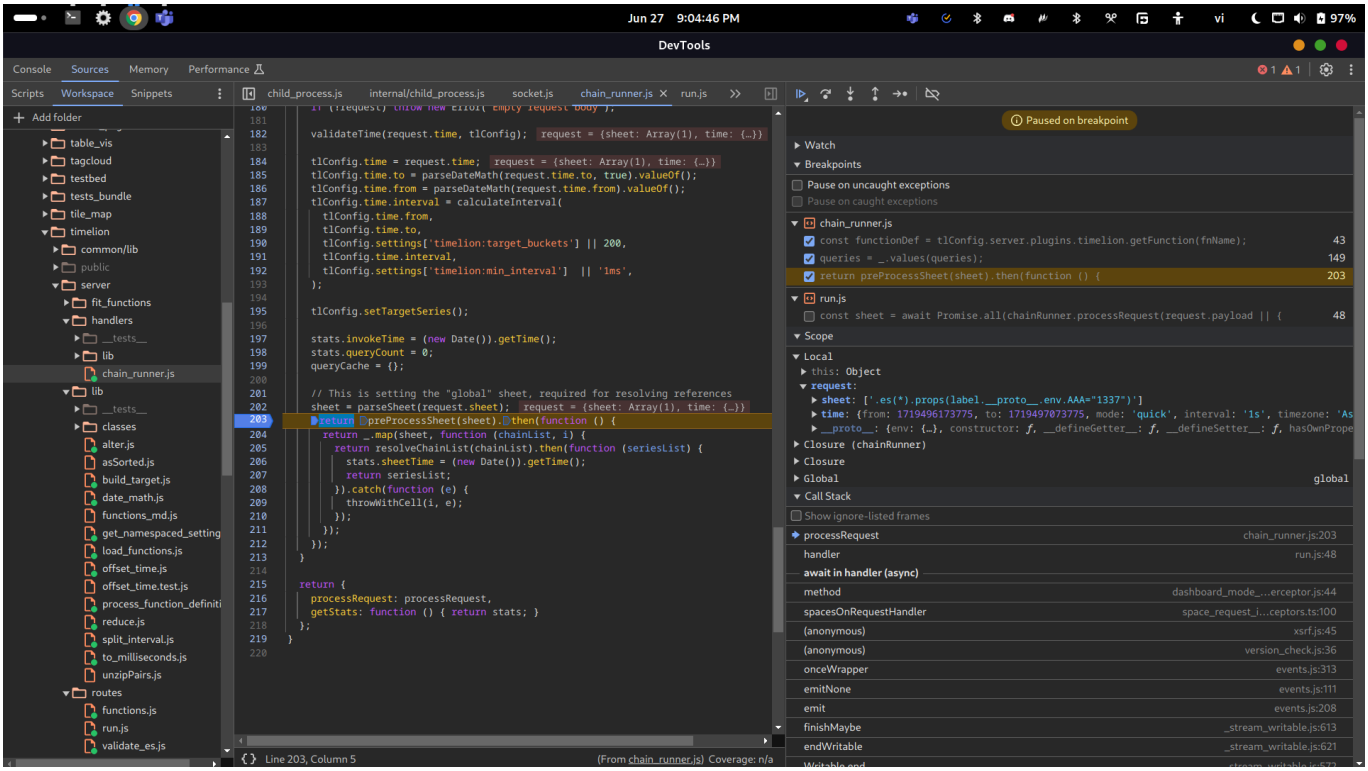
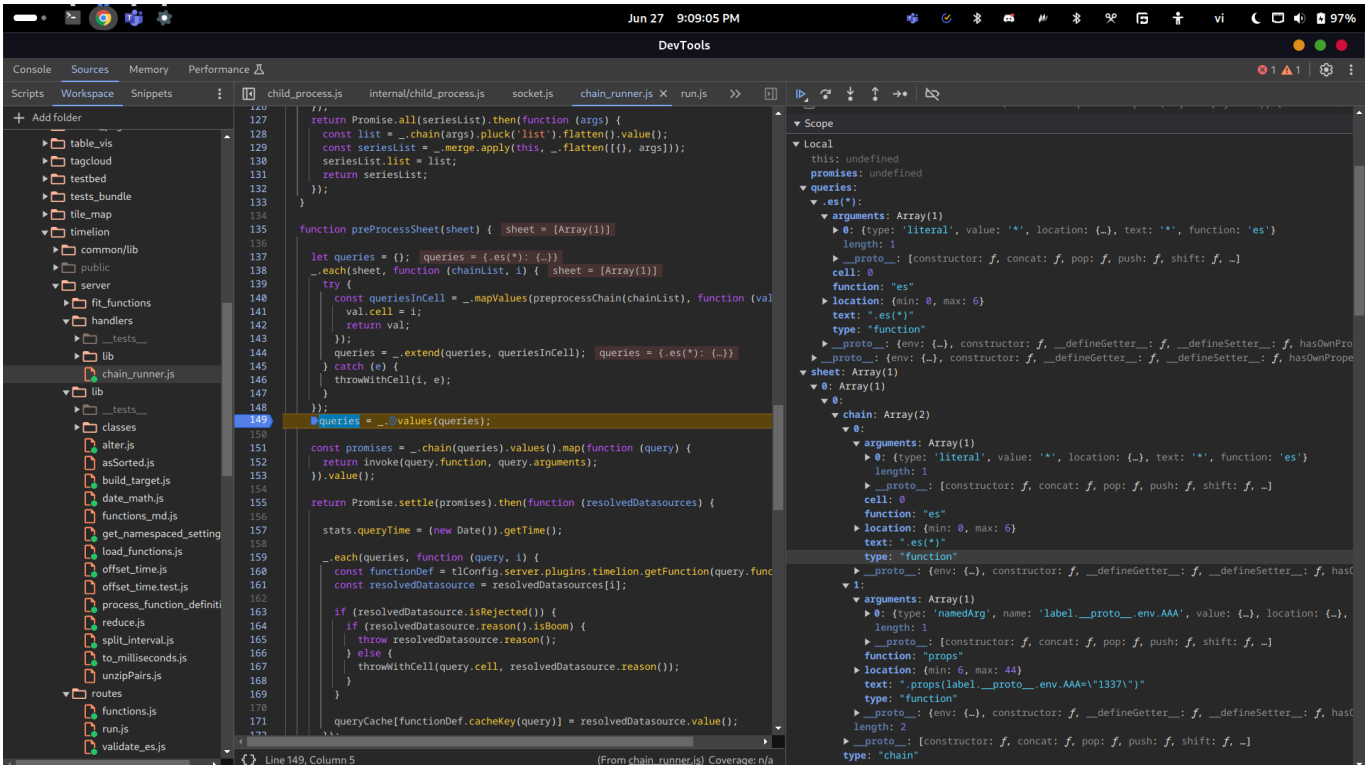
After parsing sheet:

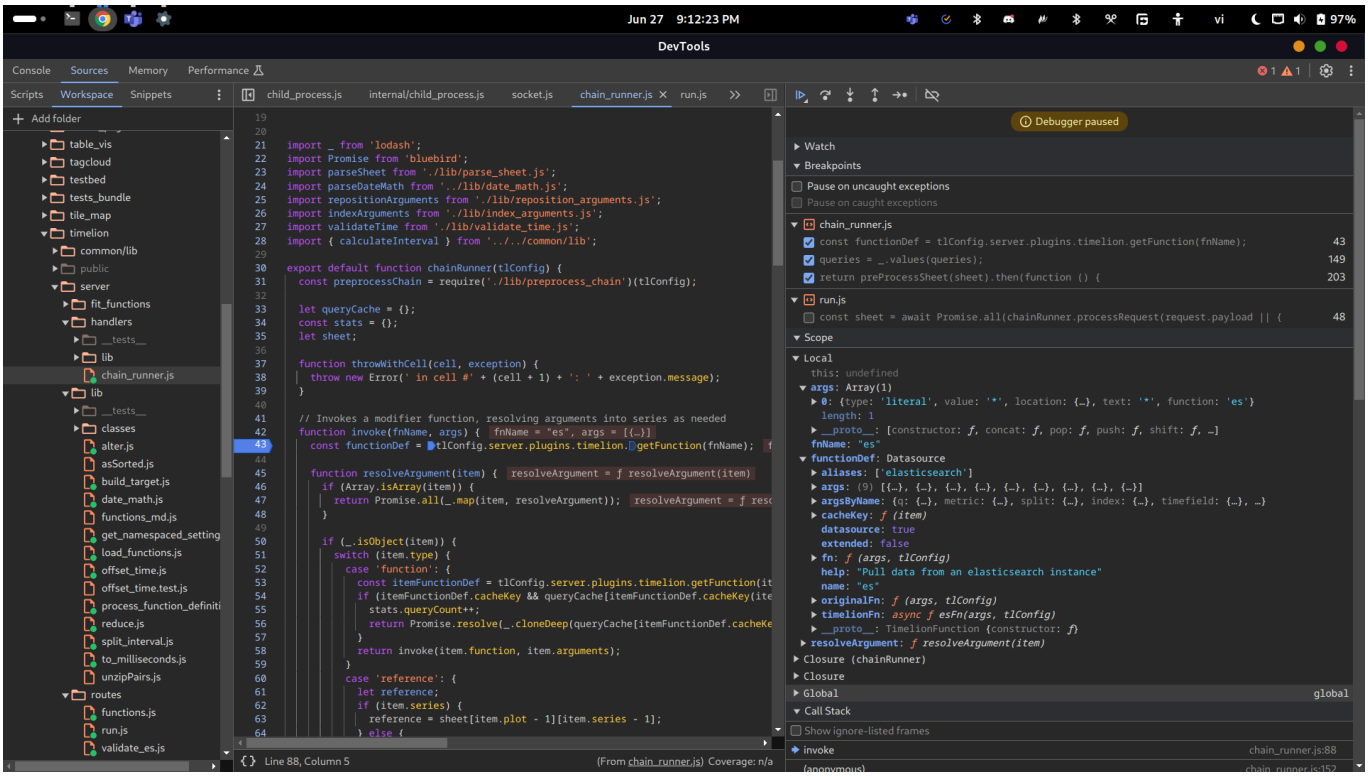


After pre-process sheet, we have a `queries` object:

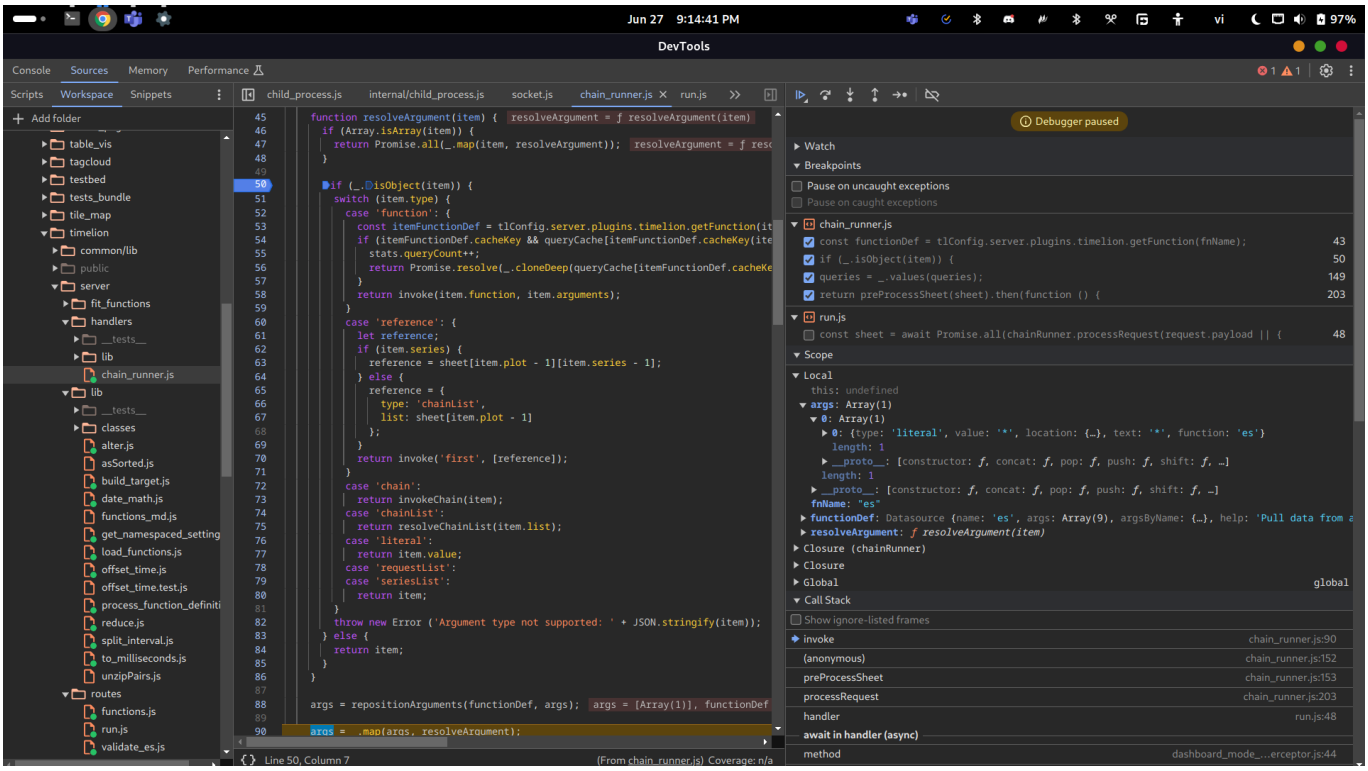


Then, we loop over the `queries` array (which is converted to an array by using `values` method) and pass into `invoke` the `query.function` and `query.arguments` for invoking the function.

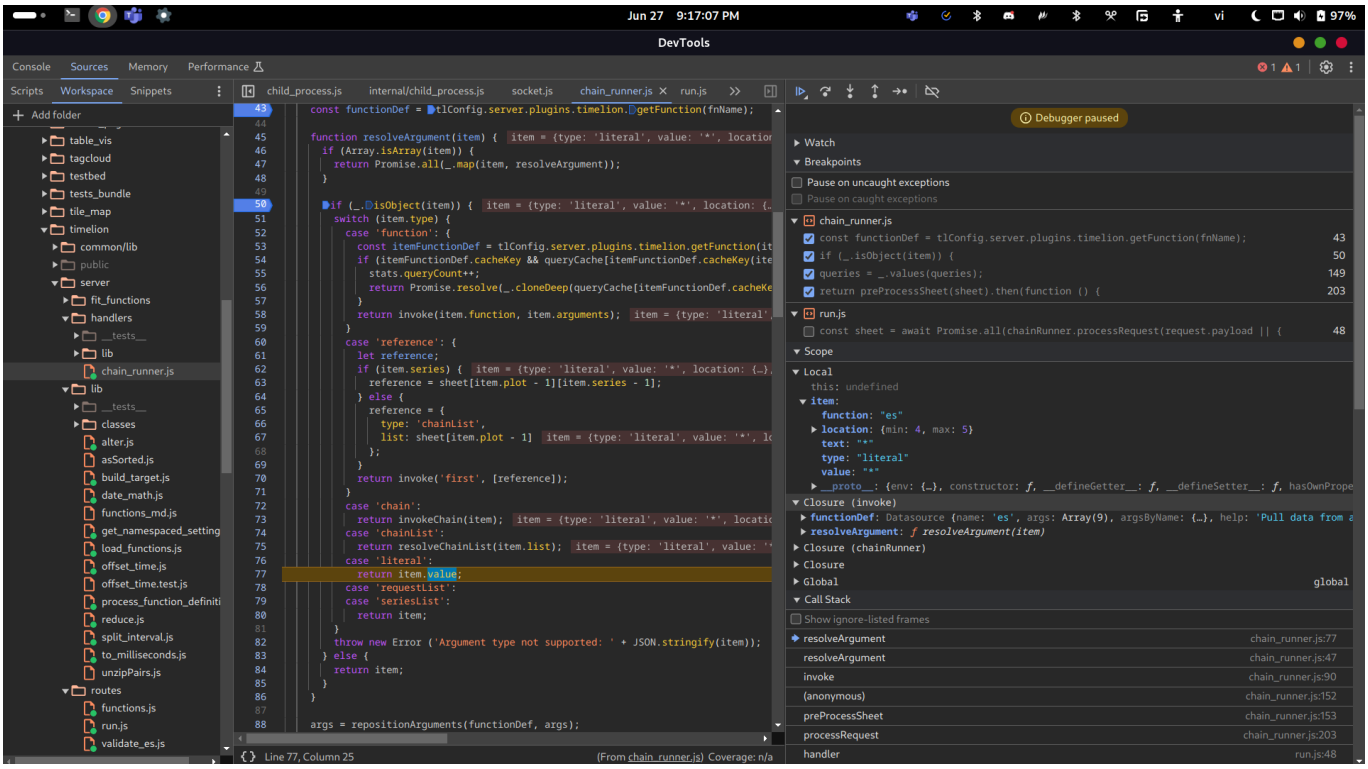
In the `invoke` function, first we get the definition of the function based on its name (`query.function`):



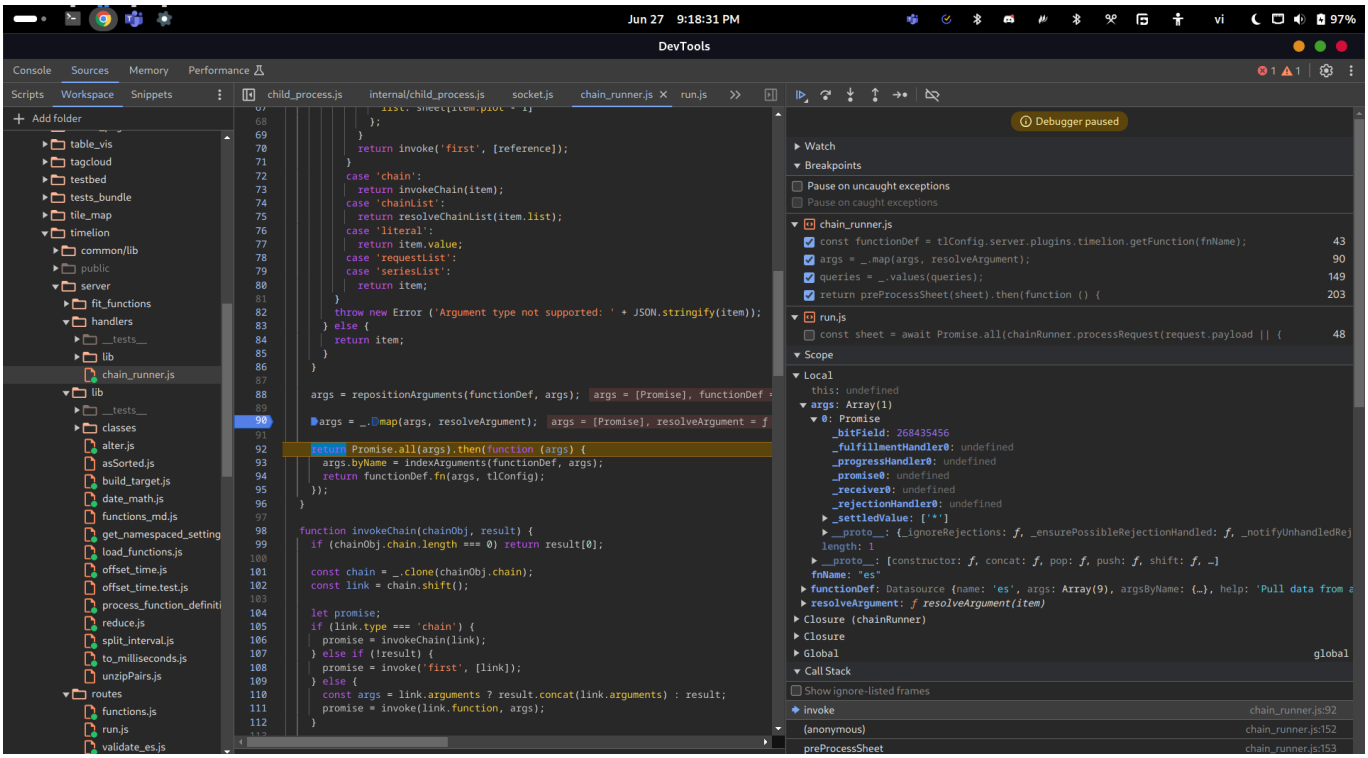
This function definition includes three types of function: `fn`, `originalFn` and `timelionFn`. Then, it loop over the `args` (which is `query.arguments`) and resolve arguments:



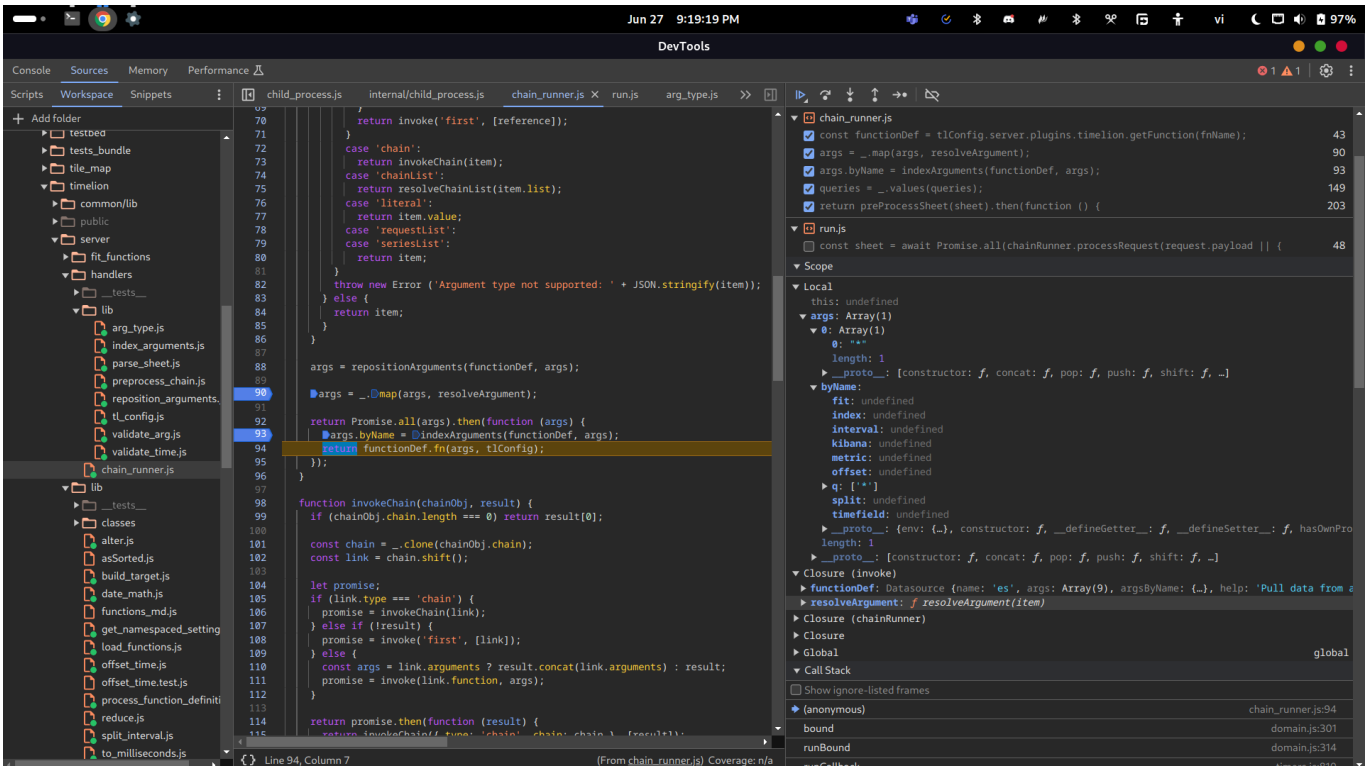
In this case, the argument in `args` is an object with `'literal'` type so it just return the literal value:



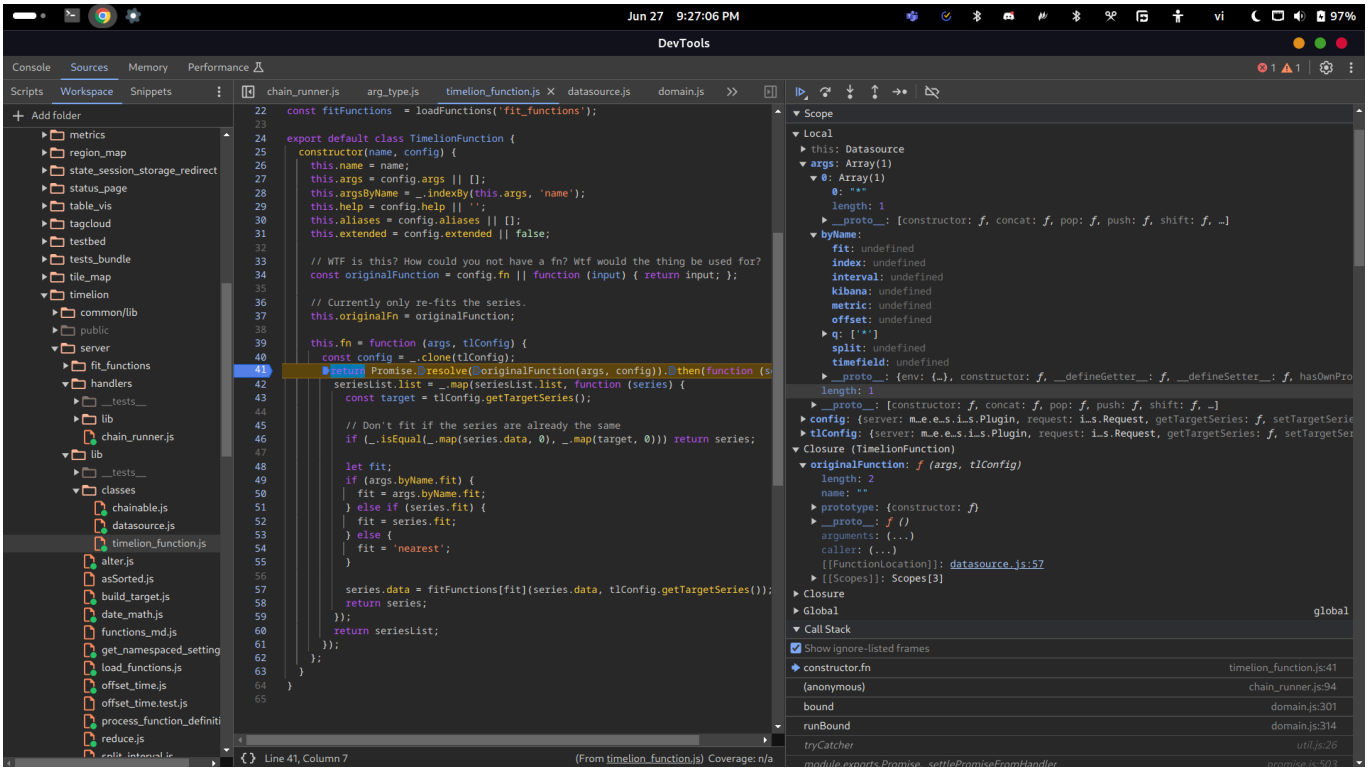
Actually, the value is wrapped in a promise.



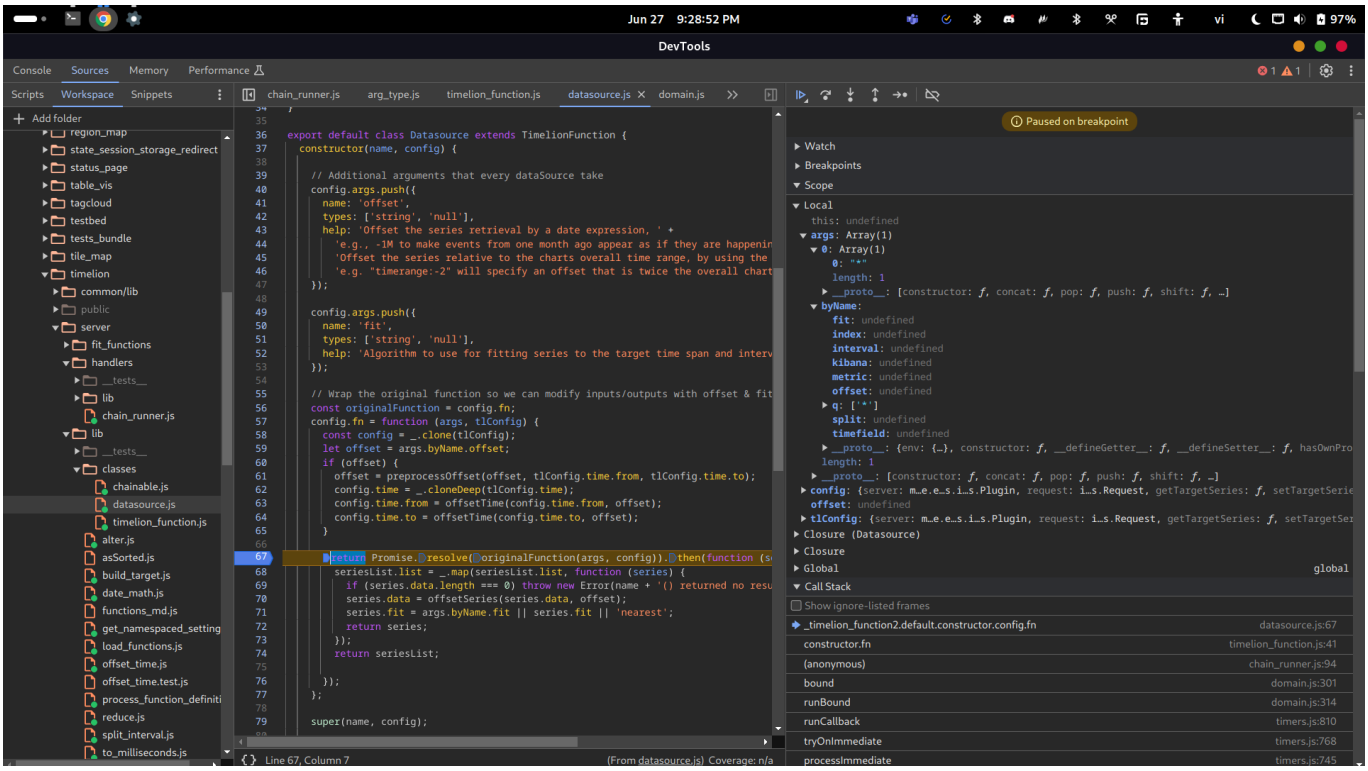
The indexed argument is an argument with `byName` object that has `q` array inside it:



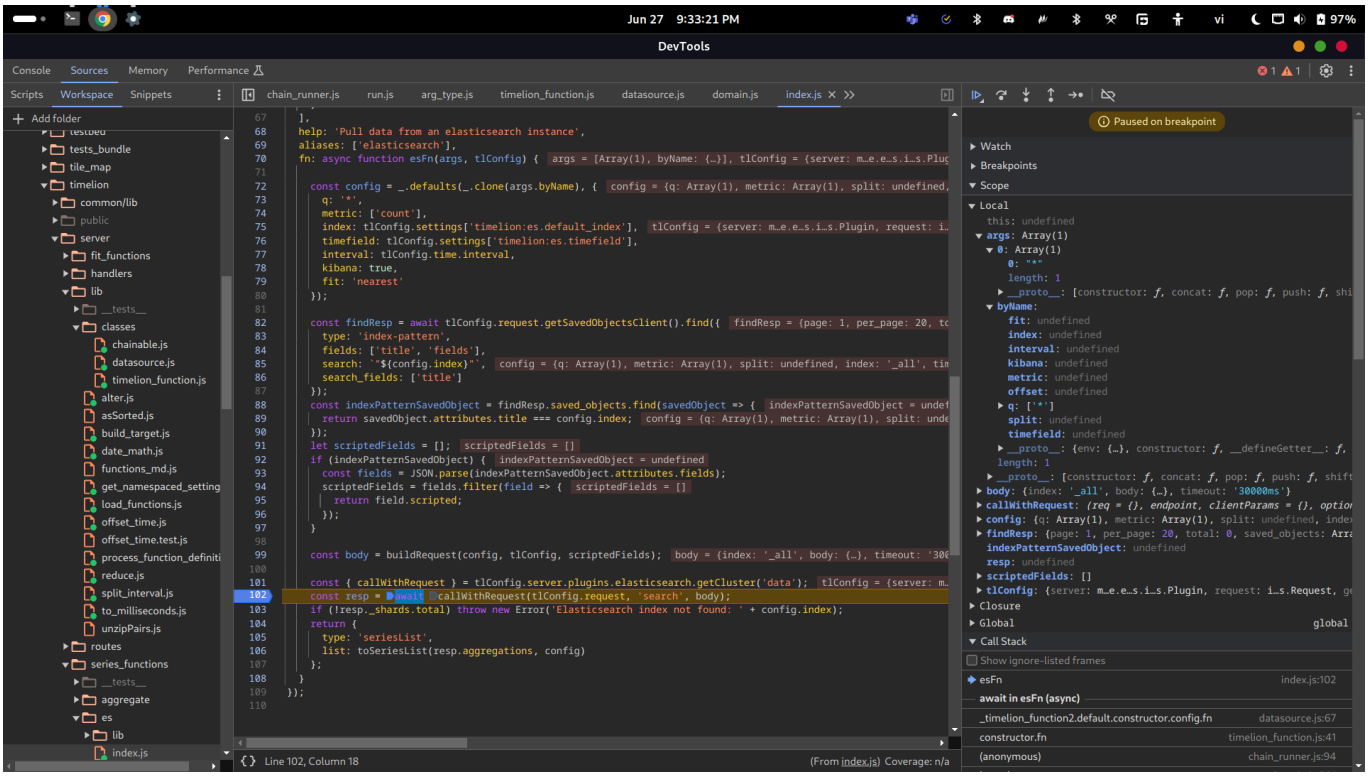
After that, the `invoke` function will call `functionDef.fn`, which is a function of the `Timelion` class, with all of the indexed arguments:



The `fn` function will call the `originalFn`, which comes from `config.fn` where `config` is `tlConfig`. And that function is `fn` of `Datasource extends Timelion` class:

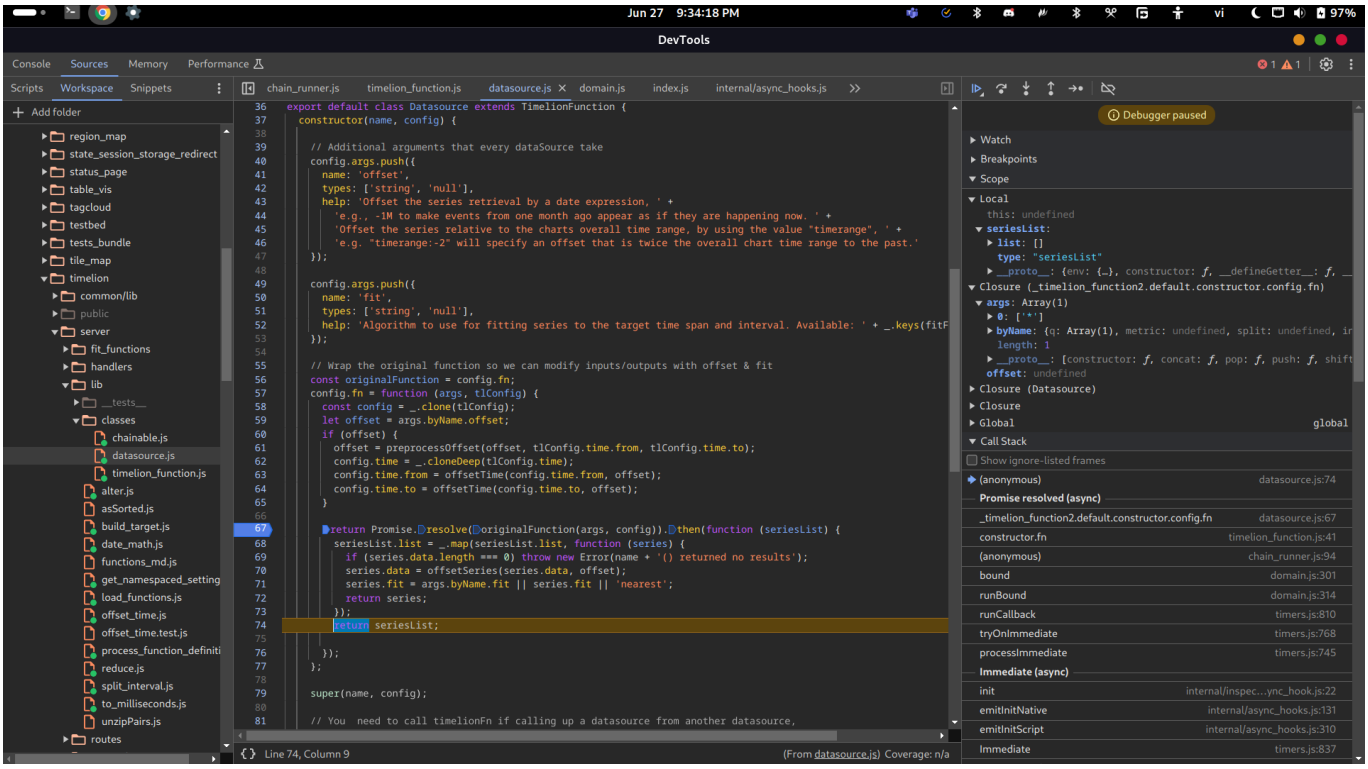


As we can see, it calls to another `originalFn` named `esFn` which returns a `seriesList` object:

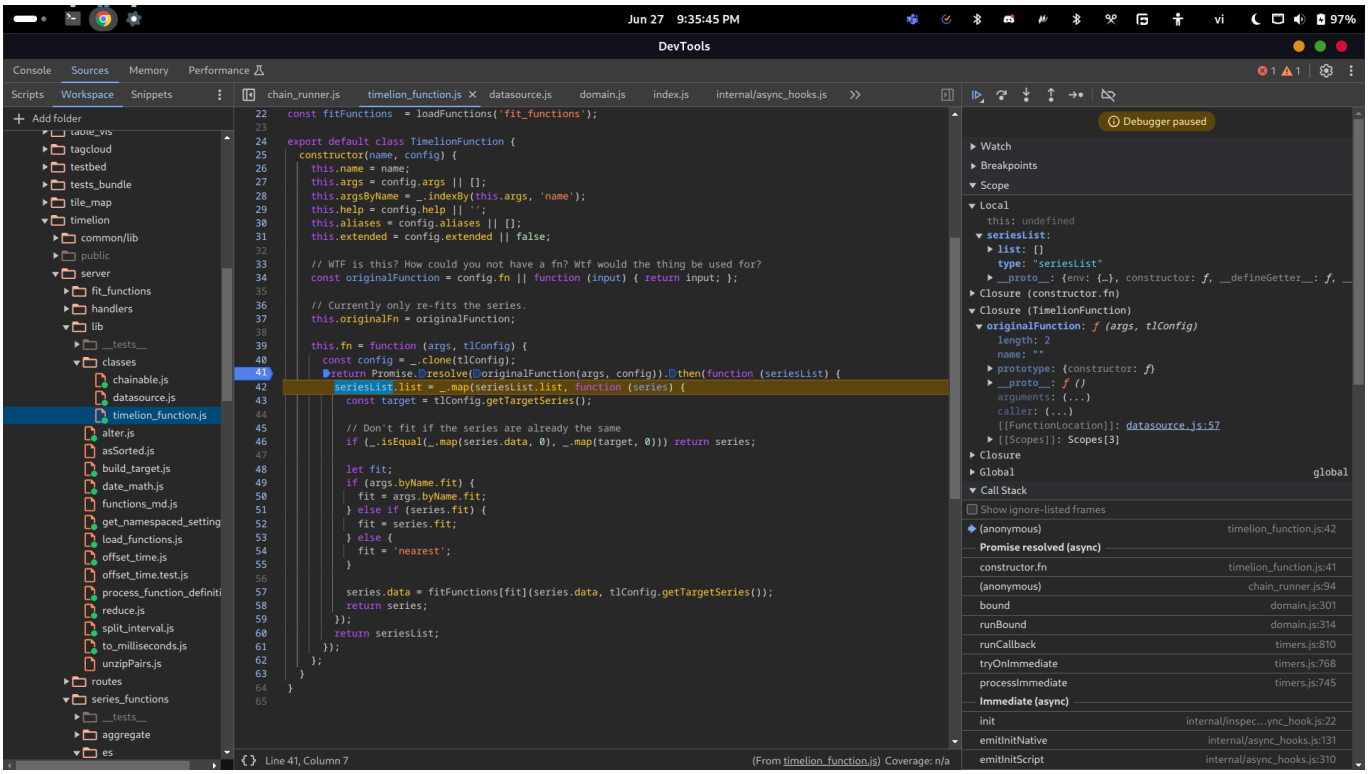


That function is used for "Pull data from an elasticsearch instance" by calling `callWithRequest` function.

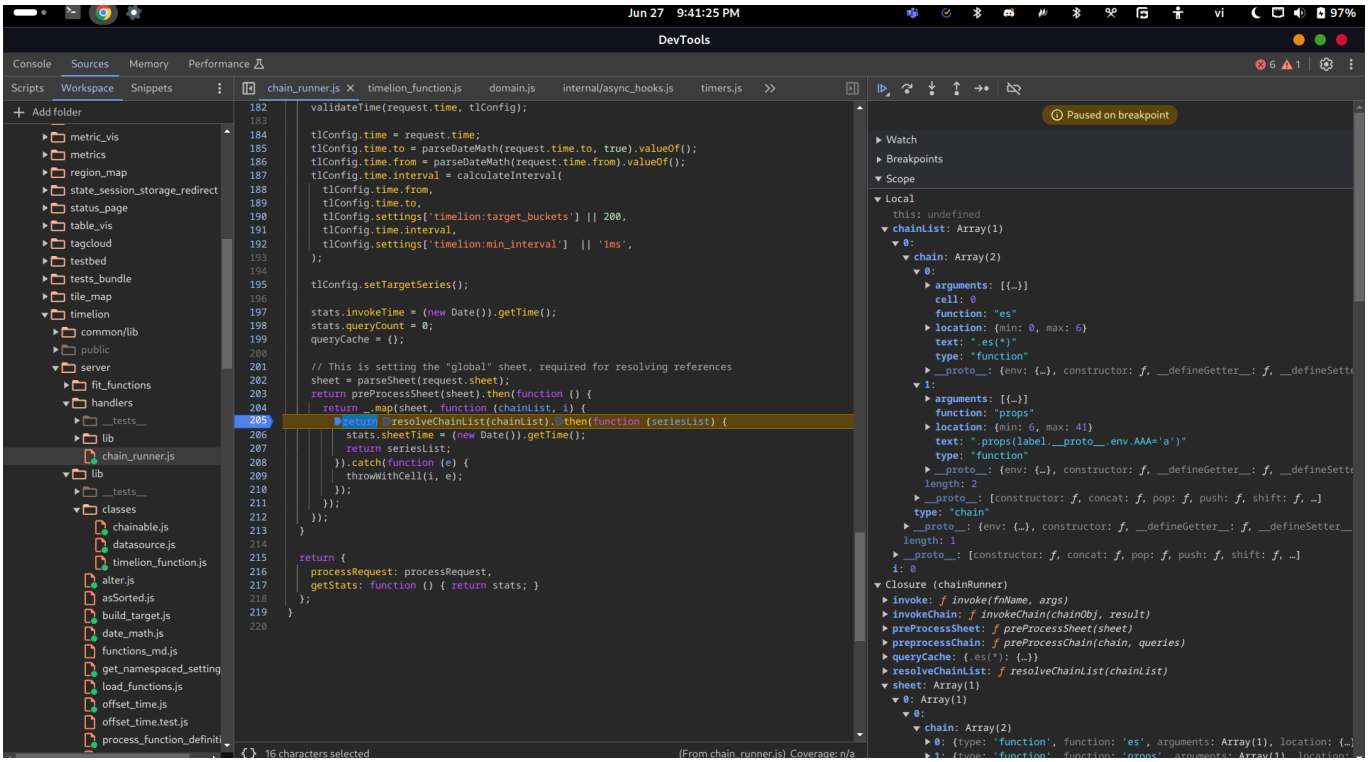
The `seriesList` looks like this in `Datasource.fn`:



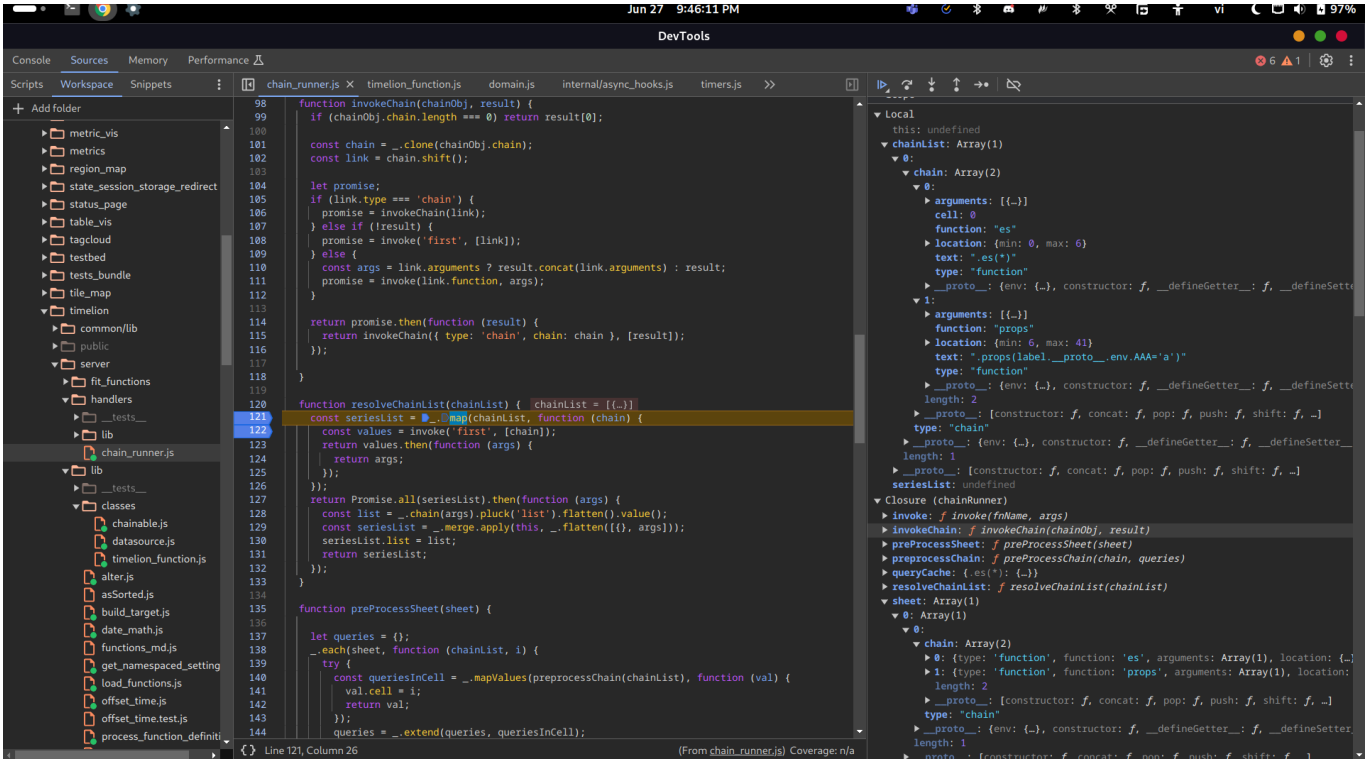
And it looks like this in `Timelion.fn`:



After pre-process sheet, the chain list is:

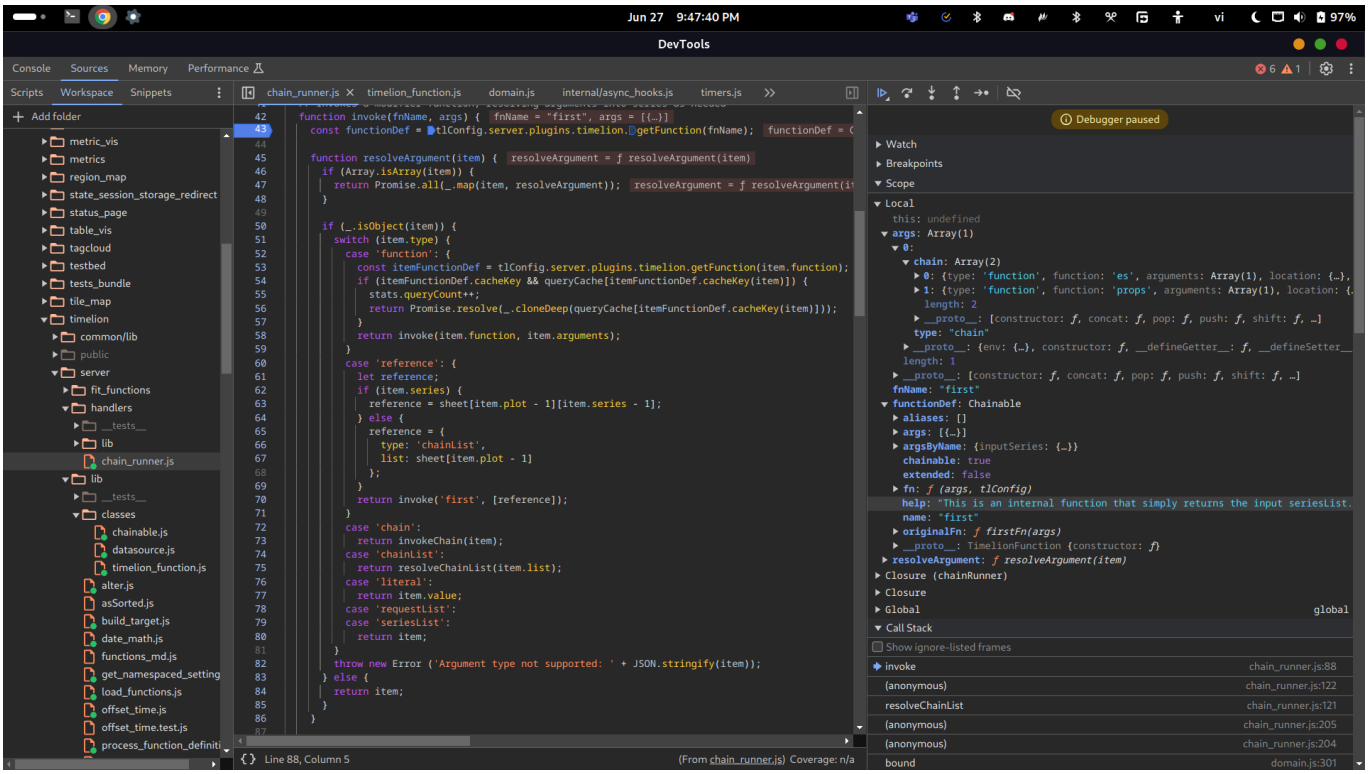


And the code will call the `resolveChainList` for resolving the chain list:



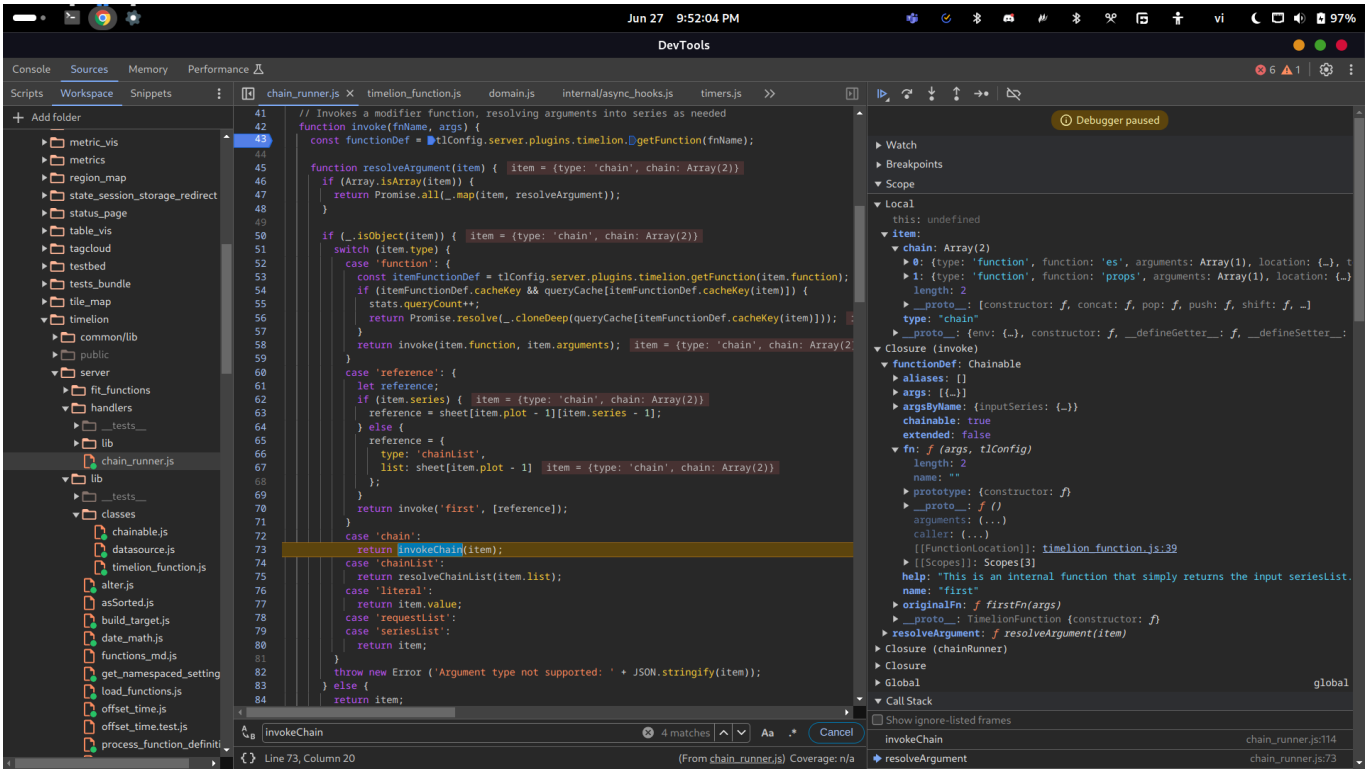
Each chain will be mapped to a Promise that is returned from a function that `invoke` with `'first'` as `fnName` and the chain itself as `args`.

The function definition is get by the name again:

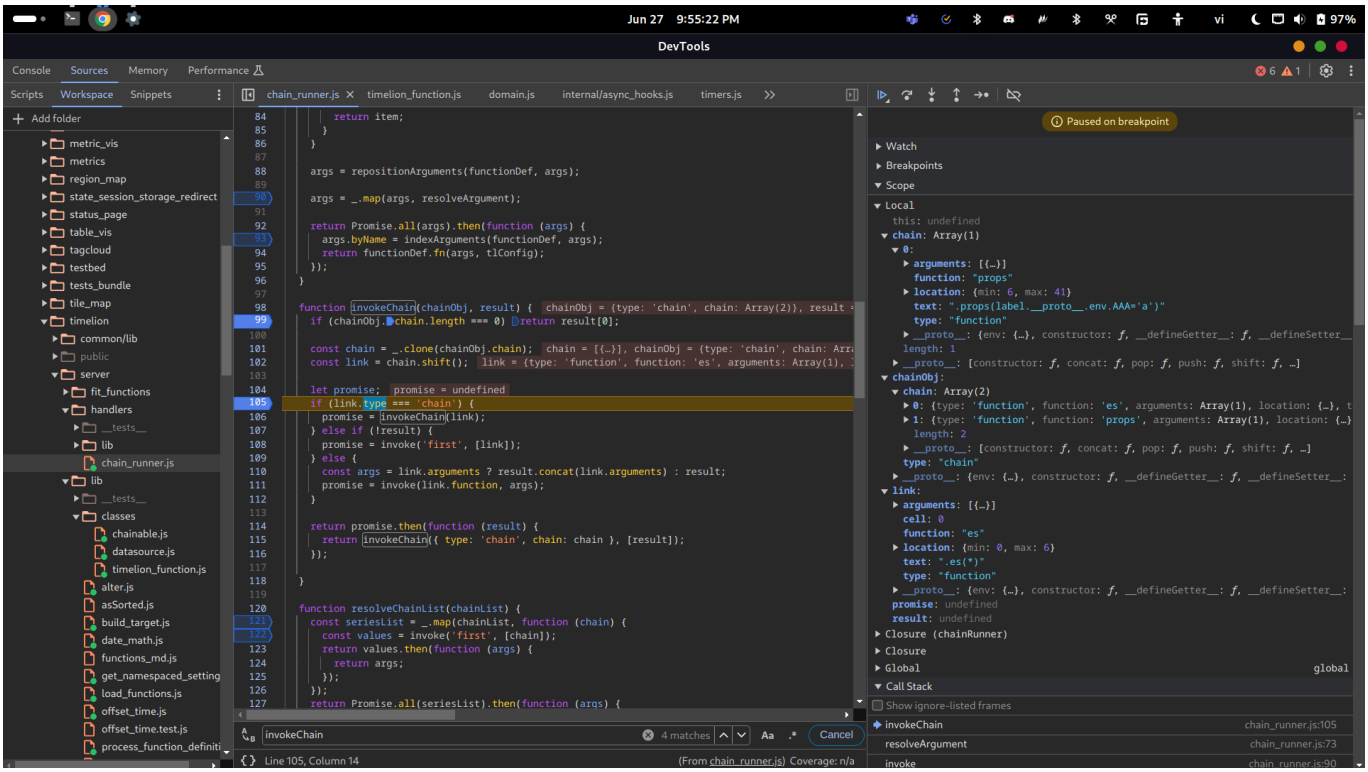


This time, `resolveArguments` receive `args` as an array and it will call the `resolveArguments` recursively for resolving each argument. But, we have only one argument with index `0`. And that argument is an object with `type == chain`.

So, in the switch case, it will call the `invokeChain` function and pass the chain object into it:

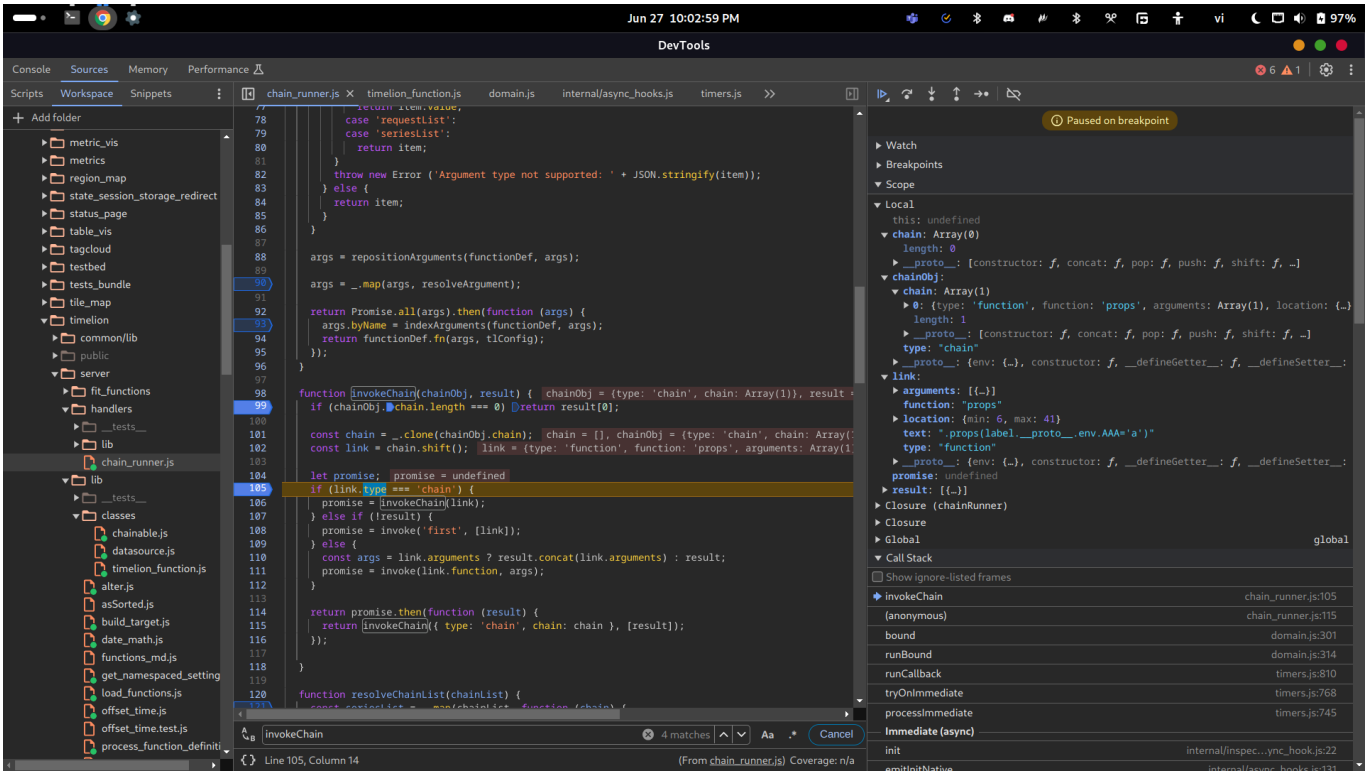


Inside the invoke chain, it will `shift` the `chain` array. The `shift` function will return the removed element so `link` will be `es` object:

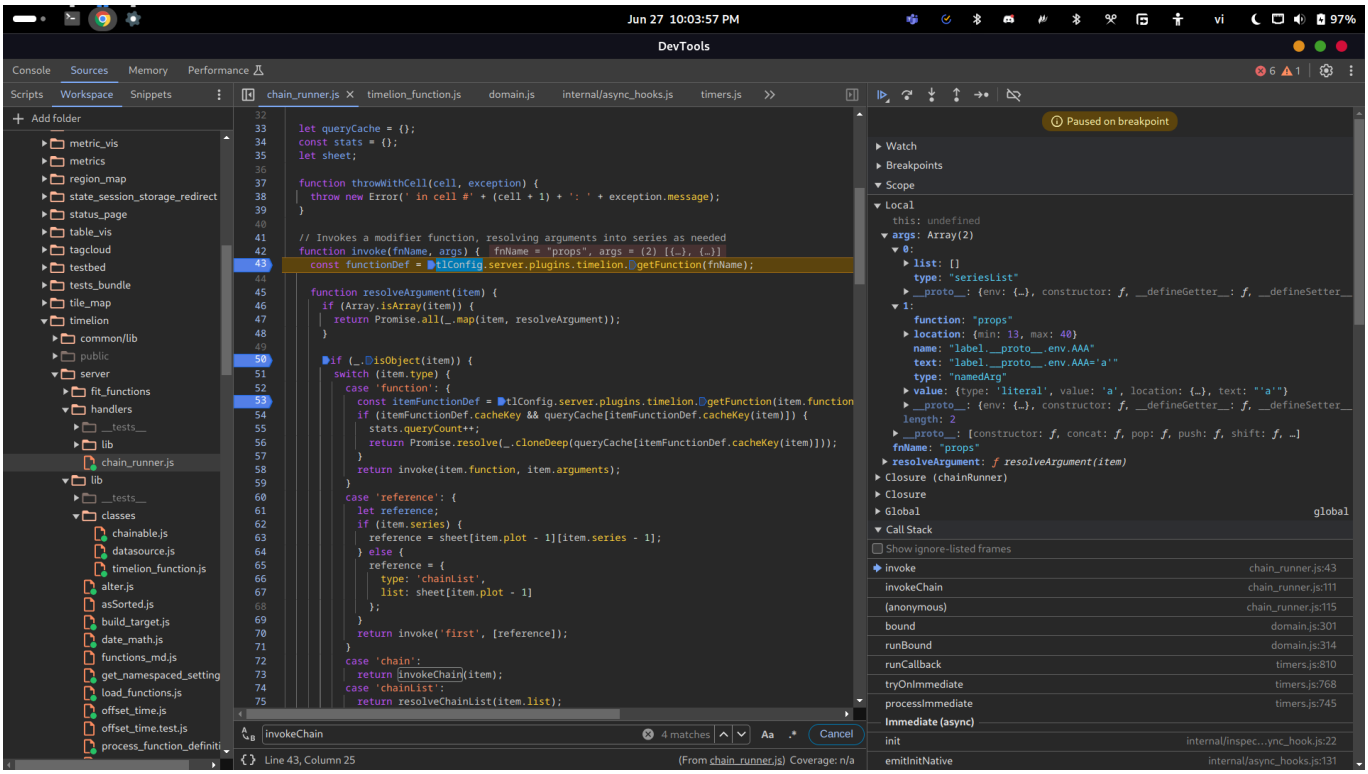


Because the type of `es` is not `'chain'` (it is `'function'`) and `!result` is truthy, the code calls the `invoke` function with `'first'` as `fnName` the `es` object as `args`:

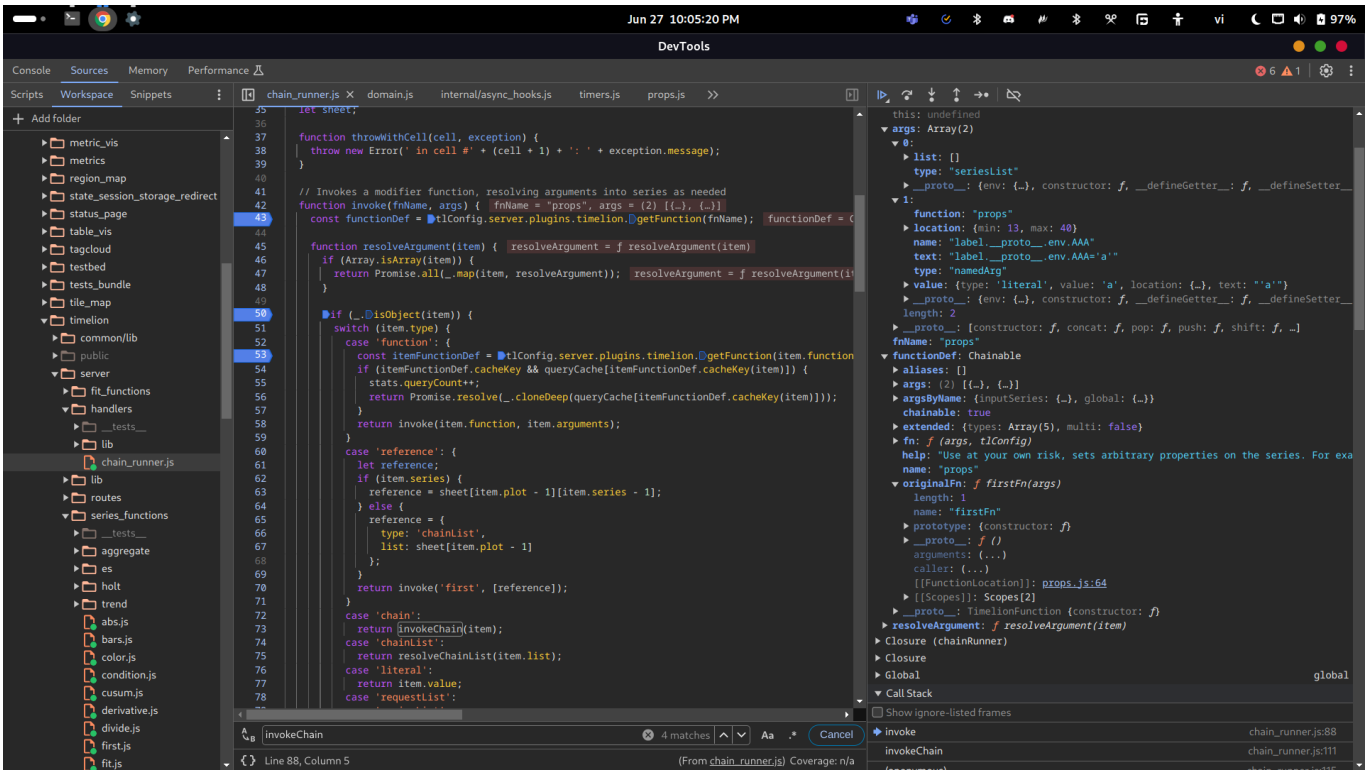
This time, the procedure is the same as above so skip to the call of `invokeChain` with `link` is `props` object:



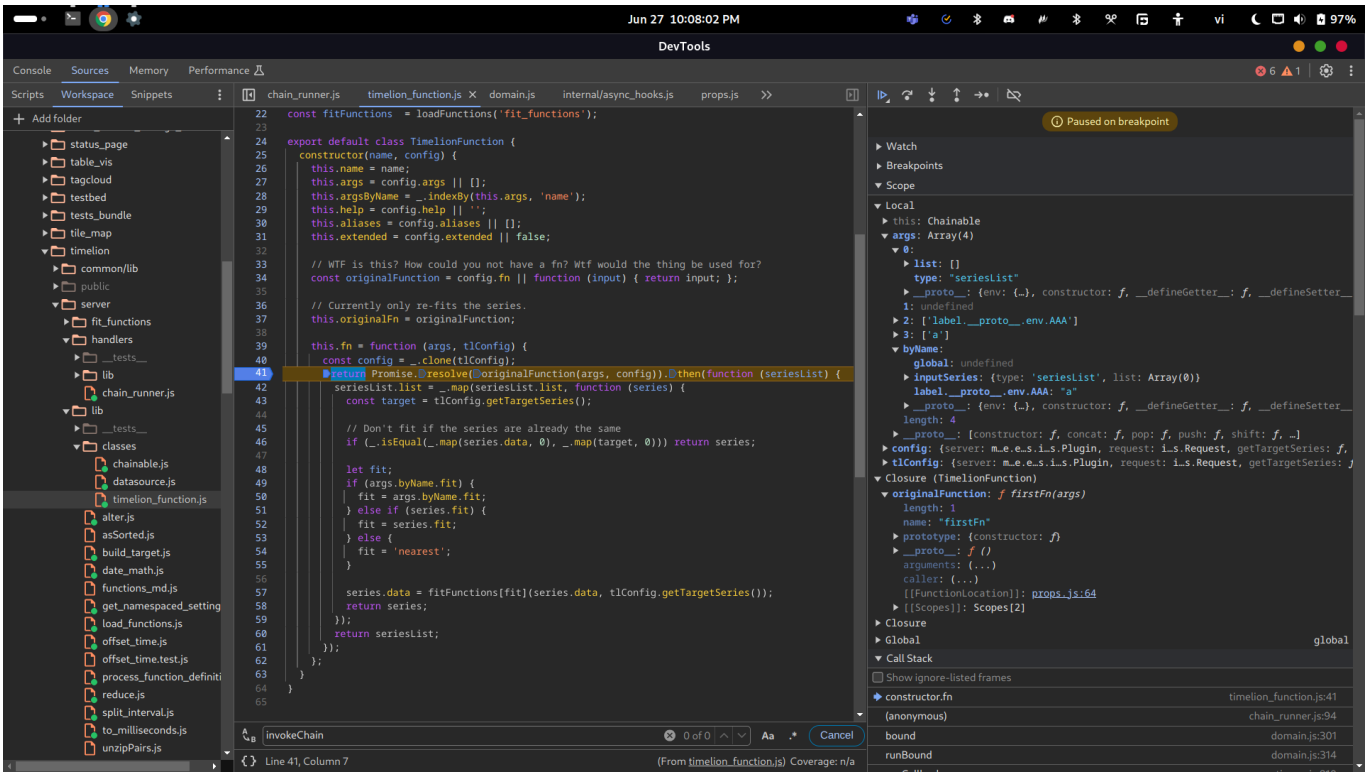
Again, the code calls the `invoke` function with `'first'` as `fnName` the `props` object as `args`:



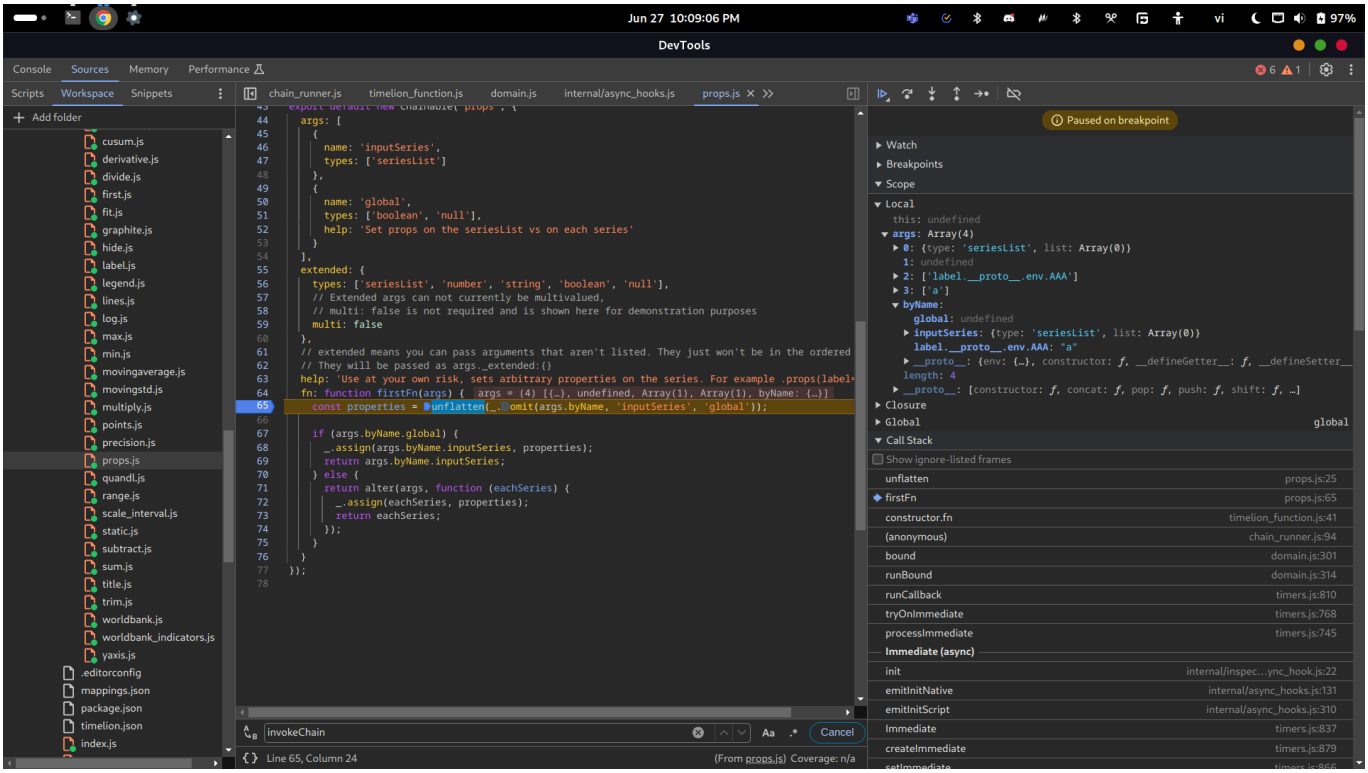
The `originalFn` of `props` now is `firstFn` instead of `fn` like `es` and only accept one argument:



That original function is will eventually call the vulnerable `unflatten` function. Actually, it will call the `fn` of `Timelion` first. Then `fn` will call the `originalFn` of `props`:



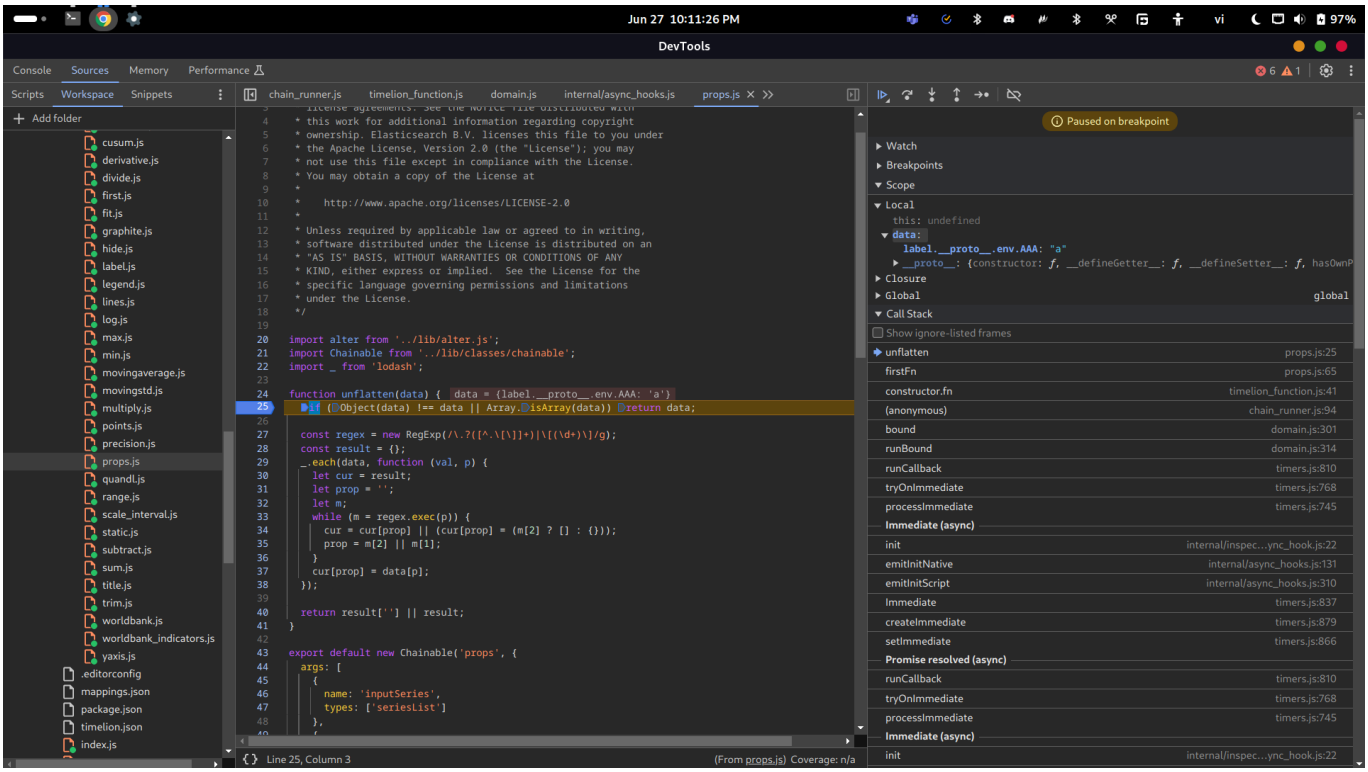
The `firstFn` of `props` will omit the `inputSeries` and `global` properties in `byName` object of `args`:



So the `data` passed into `unflatten` will be:

```
{
    label.__proto__.env.AAA: "a"
}
```

Proof:


Jun 27 10:11:26 PM
97%
DevTools
Console
Sources
Memory
Performance
Scripts
Workspace
Snippets
Add folder
chain_runner.js
timelion_function.js
domain.js
internal/async_hooks.js
props.js
cusum.js
derivative.js
divide.js
first.js
fit.js
graphite.js
hide.js
label.js
legend.js
lines.js
log.js
max.js
min.js
movingaverage.js
movingstd.js
multiply.js
points.js
precision.js
props.js
quandl.js
range.js
scale_interval.js
static.js
subtract.js
sum.js
title.js
trim.js
worldbank.js
worldbank_indicators.js
yaxis.js
.editorconfig
mappings.json
package.json
timelion.json
index.js
* this work for additional information regarding copyright
* ownership. Elasticsearch B.V. licenses this file to you under
* the Apache License, Version 2.0 (the "License"); you may
* not use this file except in compliance with the License.
* You may obtain a copy of the License at
* http://www.apache.org/licenses/LICENSE-2.0
* Unless required by applicable law or agreed to in writing,
* software distributed under the License is distributed on an
* "AS IS" BASIS, WITHOUT WARRANTIES OR CONDITIONS OF ANY
* KIND, either express or implied. See the License for the
* specific language governing permissions and limitations
* under the License.
*/
import alter from '../lib/alter.js';
import Chainable from '../lib/classes/chainable';
import _ from 'lodash';
function unflatten(data) { data = {label.__proto__.env.AAA: 'a'}
if (Object(data) !== data || Array.isArray(data)) return data;
const regex = new RegExp(/\.?([^.\[\]]+)|\[(\d+)\]/g);
const result = {};
_.each(data, function (val, p) {
let cur = result;
let prop = '';
let m;
while (m = regex.exec(p)) {
cur = cur[prop] || (cur[prop] = (m[2] ? [] : {}));
prop = m[2] || m[1];
}
cur[prop] = data[p];
});
return result[''] || result;
}
export default new Chainable('props', {
args: [
{
name: 'inputSeries',
types: ['seriesList']
},
Line 25, Column 3
(From props.js) Coverage: n/a
Paused on breakpoint
Watch
Breakpoints
Scope
Local
this: undefined
data:
label.__proto__.env.AAA: "a"
__proto__: {constructor: ƒ, __defineGetter__: ƒ, __defineSetter__: ƒ, hasOwnP
Closure
Global
global
Call Stack
Show ignore-listed frames
unflatten props.js:25
firstFn props.js:65
constructor.fn timelion_function.js:41
(anonymous) chain_runner.js:94
bound domain.js:301
runBound domain.js:314
runCallback timers.js:810
tryOnImmediate timers.js:768
processImmediate timers.js:745
Immediate (async)
init internal/inspec…ync_hook.js:22
emitInitNative internal/async_hooks.js:131
emitInitScript internal/async_hooks.js:310
Immediate timers.js:837
createImmediate timers.js:879
setImmediate timers.js:866
Promise resolved (async)
runCallback timers.js:810
tryOnImmediate timers.js:768
processImmediate timers.js:745
Immediate (async)
init internal/inspec…ync_hook.js:22