

Inspecting Node.js Performance

With N|Solid and Deoptigate

@thlorenz

NODESOURCE®

Inspecting Node.js Performance



@thlorenz

NODESOURCE®

Come to Colombia



@thlorenz

NODESOURCE®

Slides

nsrc.io/deoptigate-slides

@thlorenz

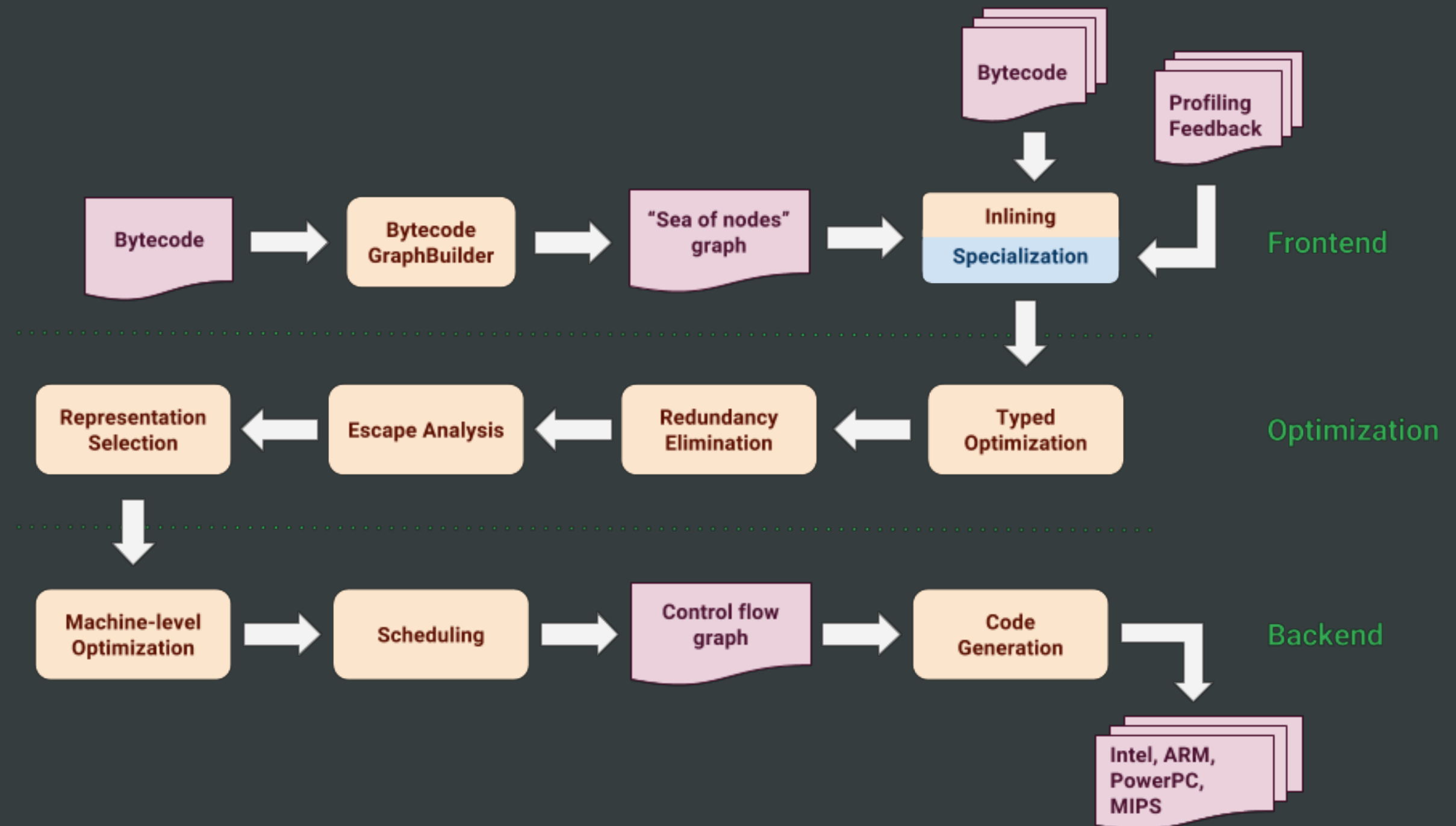
NODESOURCE®

What changed in V8 and Why

- Overview of New Compiler Pipeline
- Why is it Faster?
- Which language features are supported?
- What does that mean for your JavaScript?

New V8 Compiler Pipeline

Cleanly split into *Frontend*,
Optimization Layer and *Backend*



New V8 Compiler Pipeline

- Ignition Interpreter
- TurboFan optimizing Compiler

Ignition Interpreter

- runs highly optimized interpreter code
- *Inline Caches (aka ICs)* gather knowledge about types while program runs, stored in *FeedbackVector*

TurboFan Optimizing Compiler

- recompiles and optimizes *hot code* identified by the runtime profiler
- speculates that kinds of values seen in the past will be seen in the future and generates optimized code just for those cases

Many More Details



If interested watch this talk:

<https://youtu.be/J9HAvlW7BqA>

Just a Few More Things



How to Write Code that runs Fast?

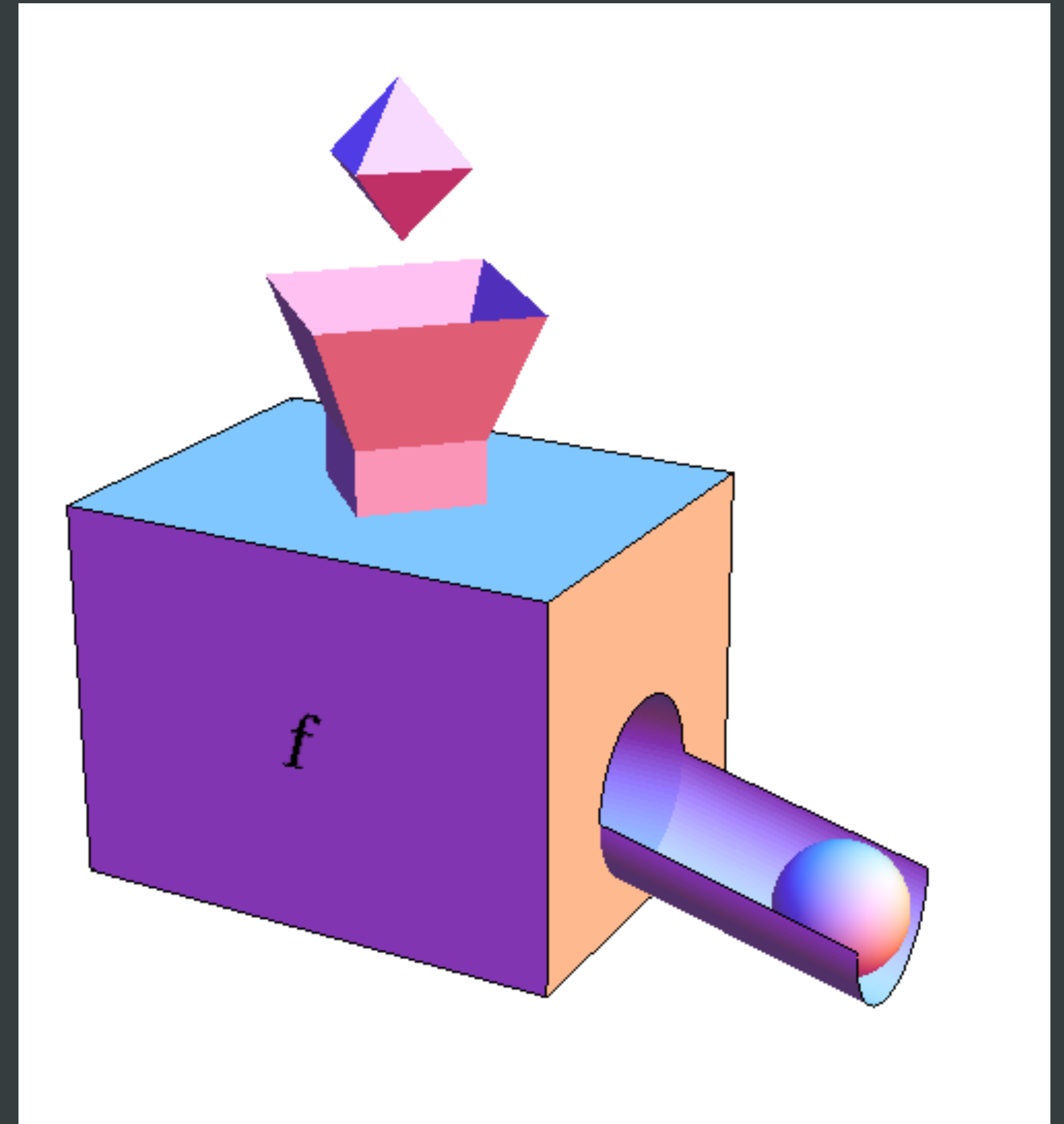
Optimized code for fewer cases is smaller and executes at peak speed.



Smaller is Better

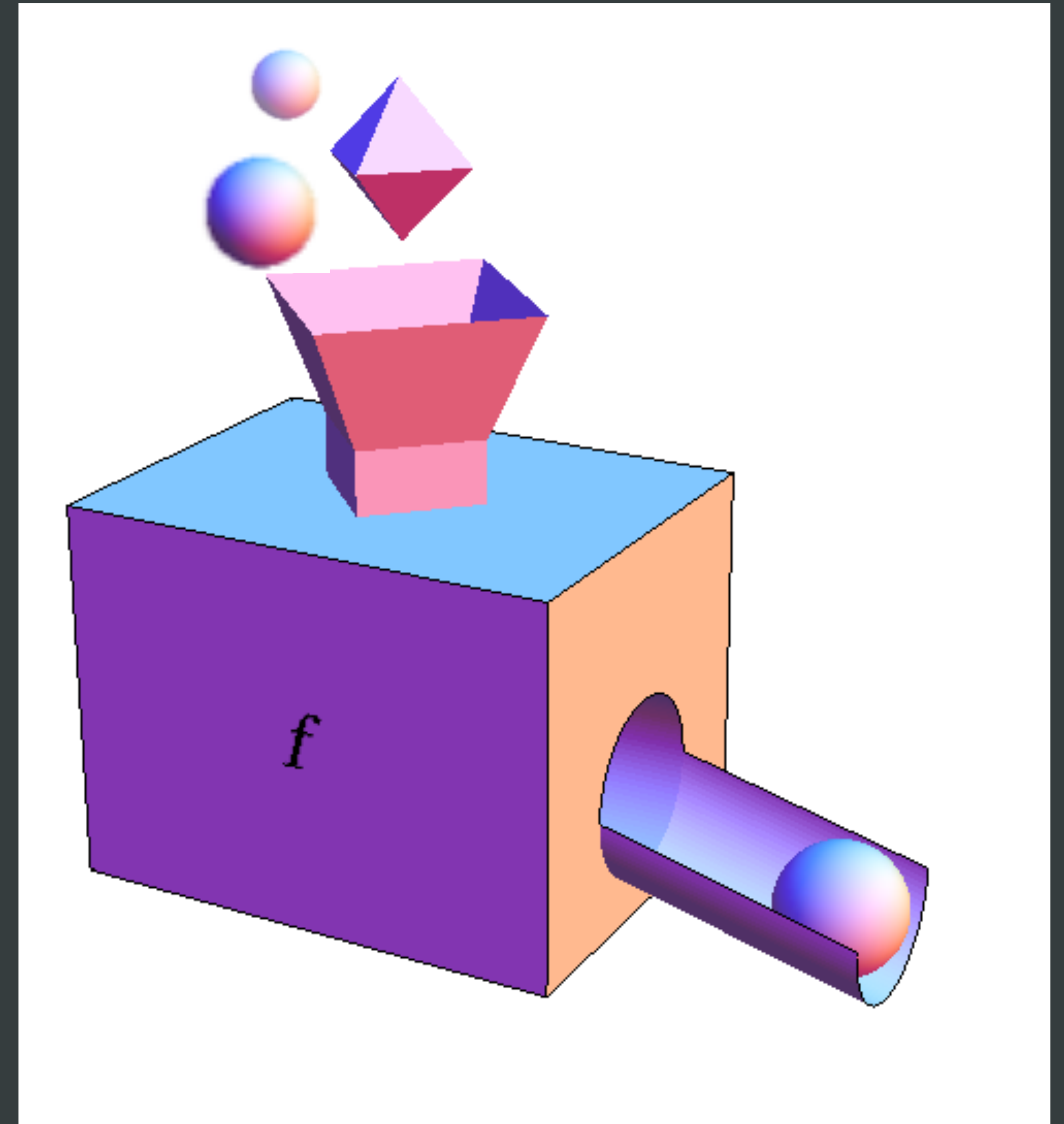
Mono-Poly-Megamorphism

Monomorphic === One Input Type



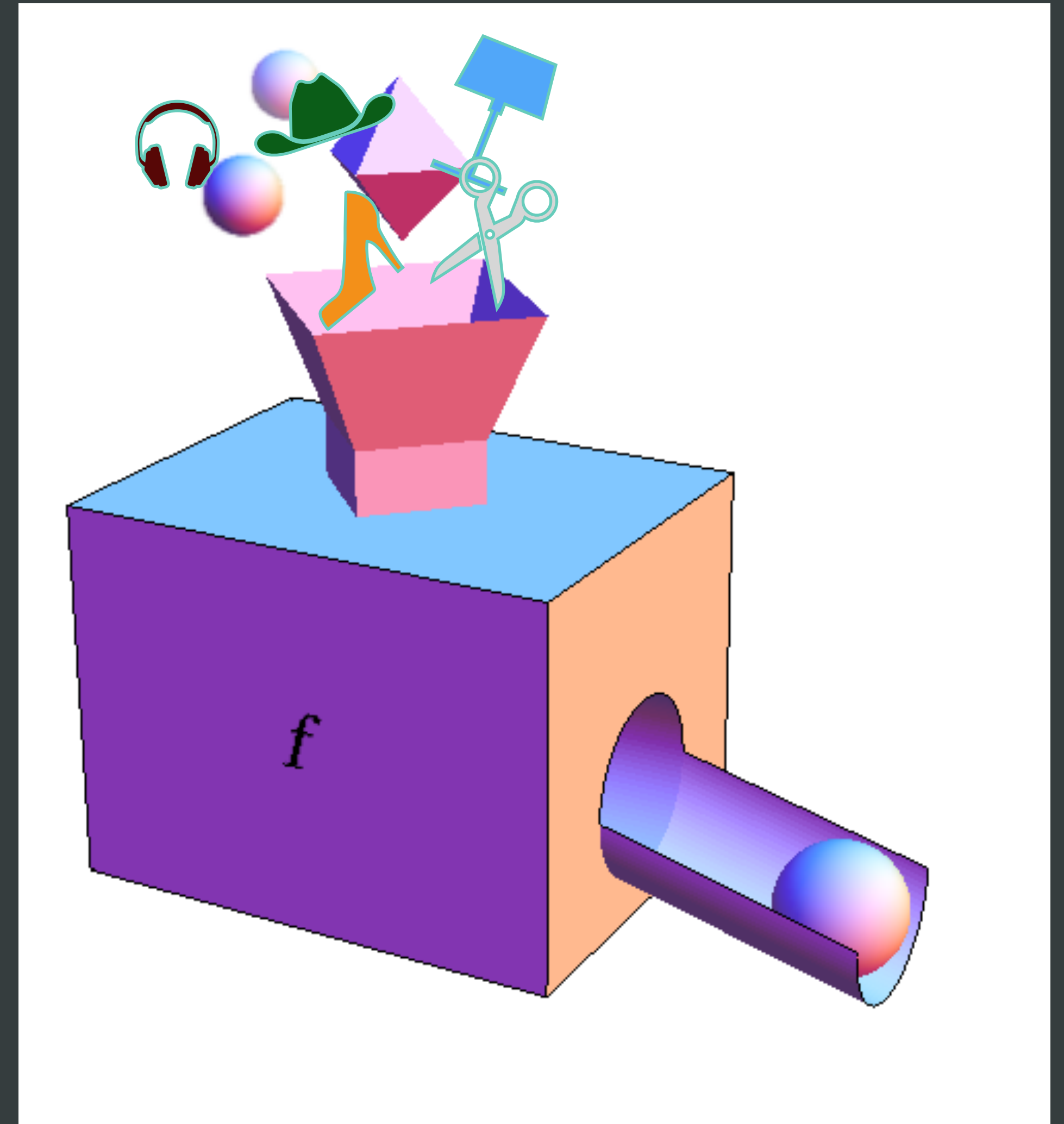
Mono-Poly-Megamorphism

Polymorphic === Two - Four Input Types



Mono-Poly-Megamorphism

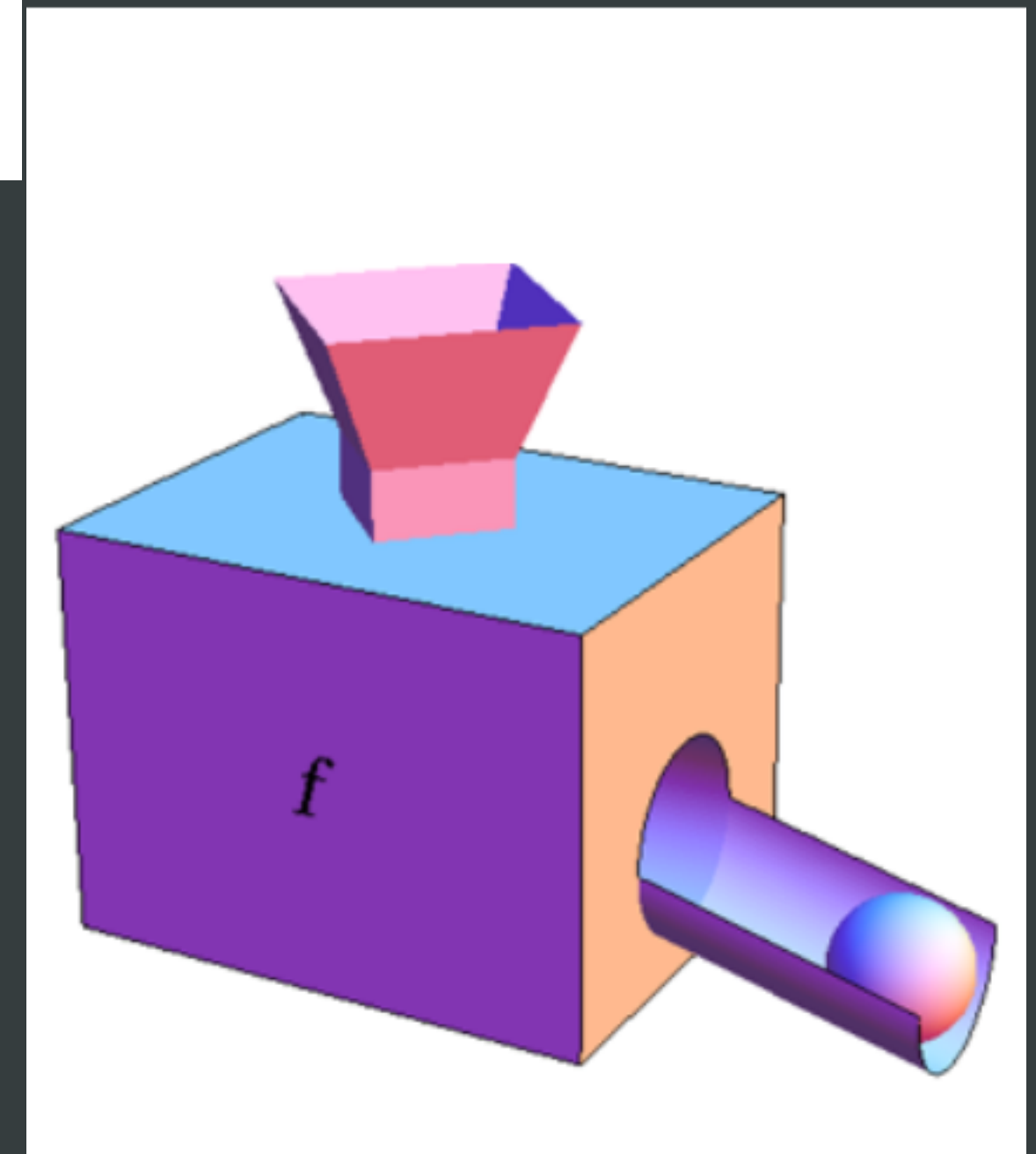
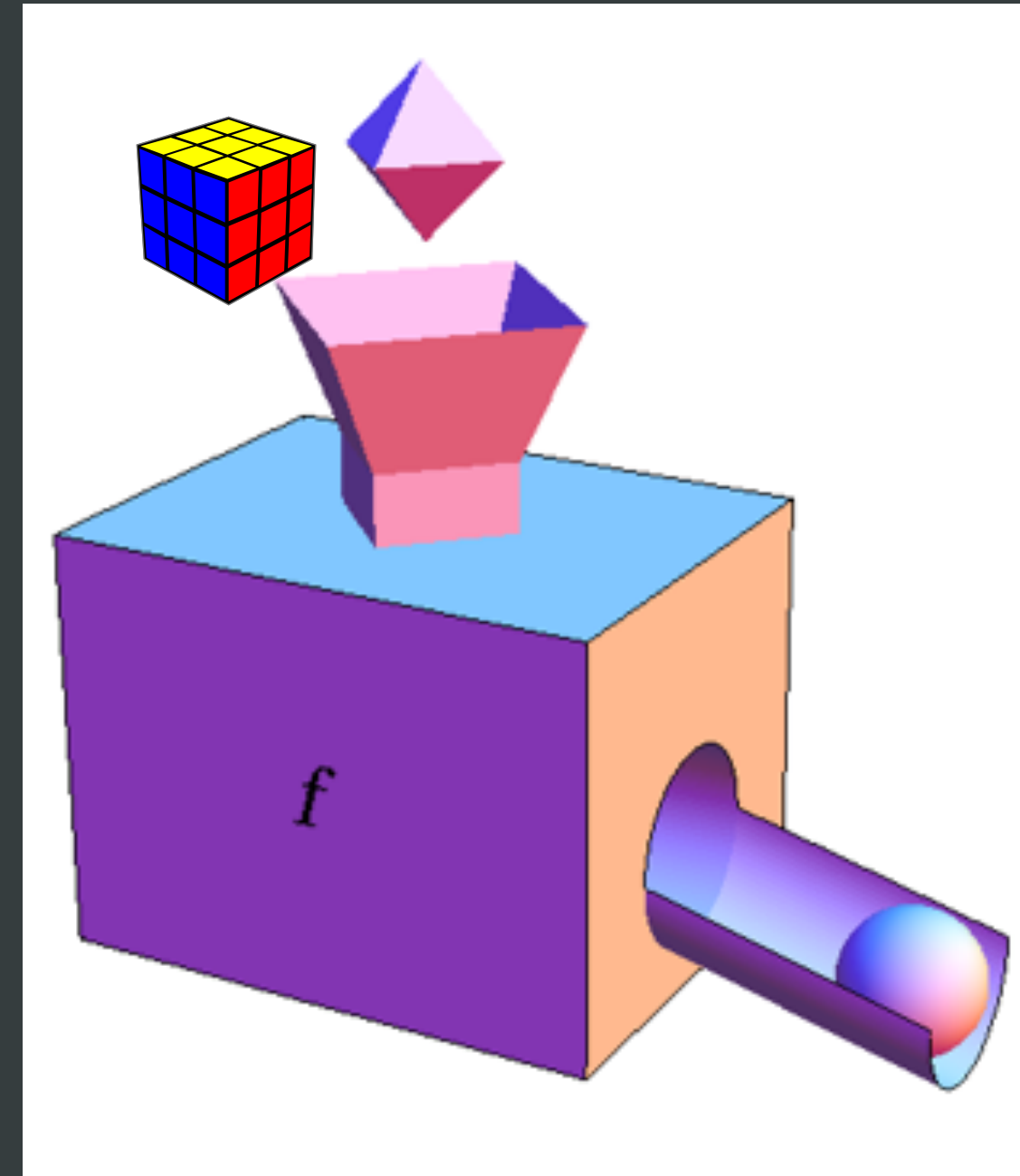
Megamorphic === Five+ Input Types



Mono-Poly-Megamorphism

Monomorphic operations are easier optimized

Prefer multiple monomorphic functions over one polymorphic function where performance matters



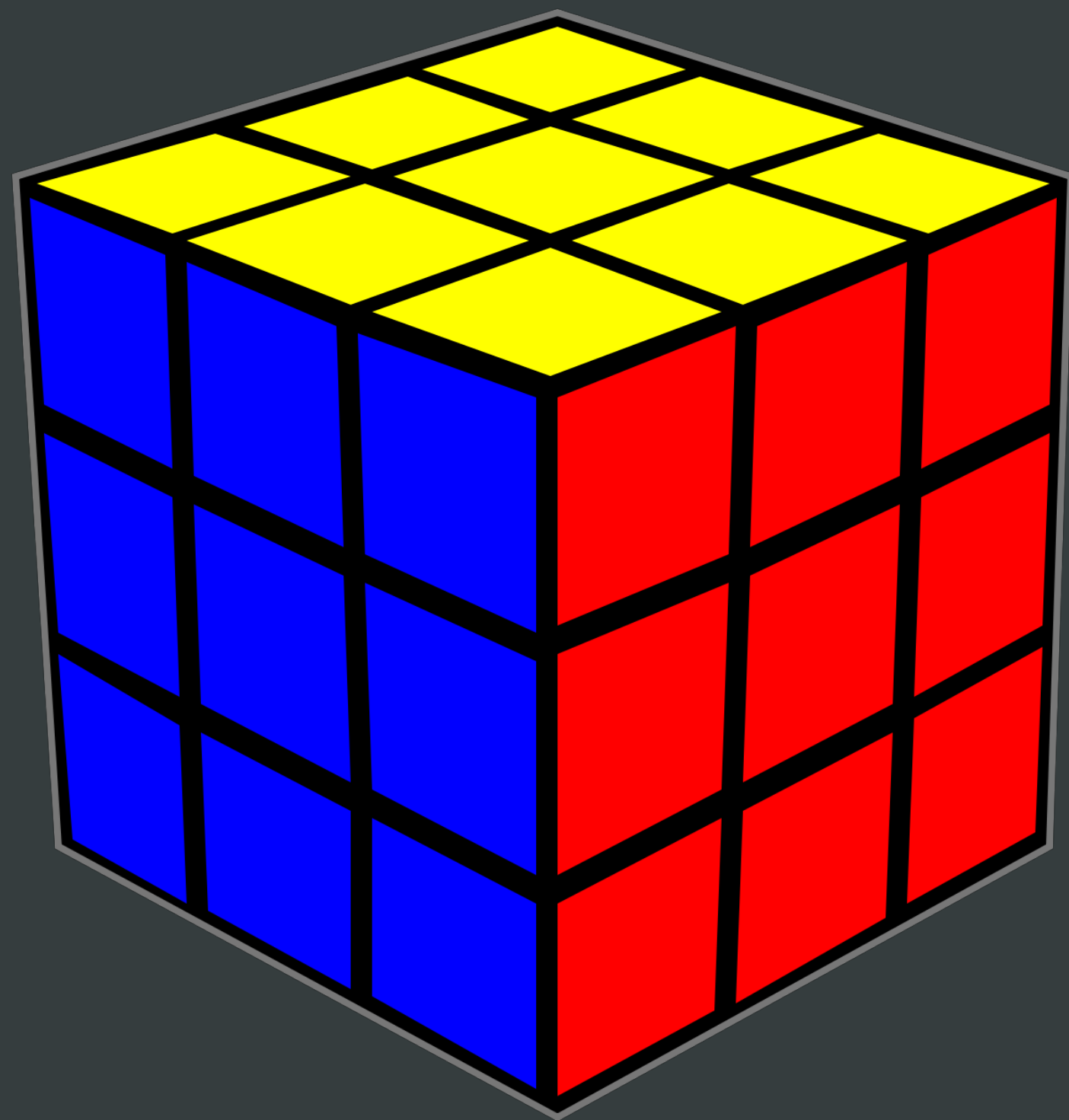
Mono-Poly-Megamorphism

V8 creates hidden classes for your
JavaScript objects *aka* Maps



Mono-Poly-Megamorphism

Maps



Recommendations

Measure *before* and *after* each optimization you apply to your code



Measure Performance Under Load

Determine where bottlenecks are in
your application.



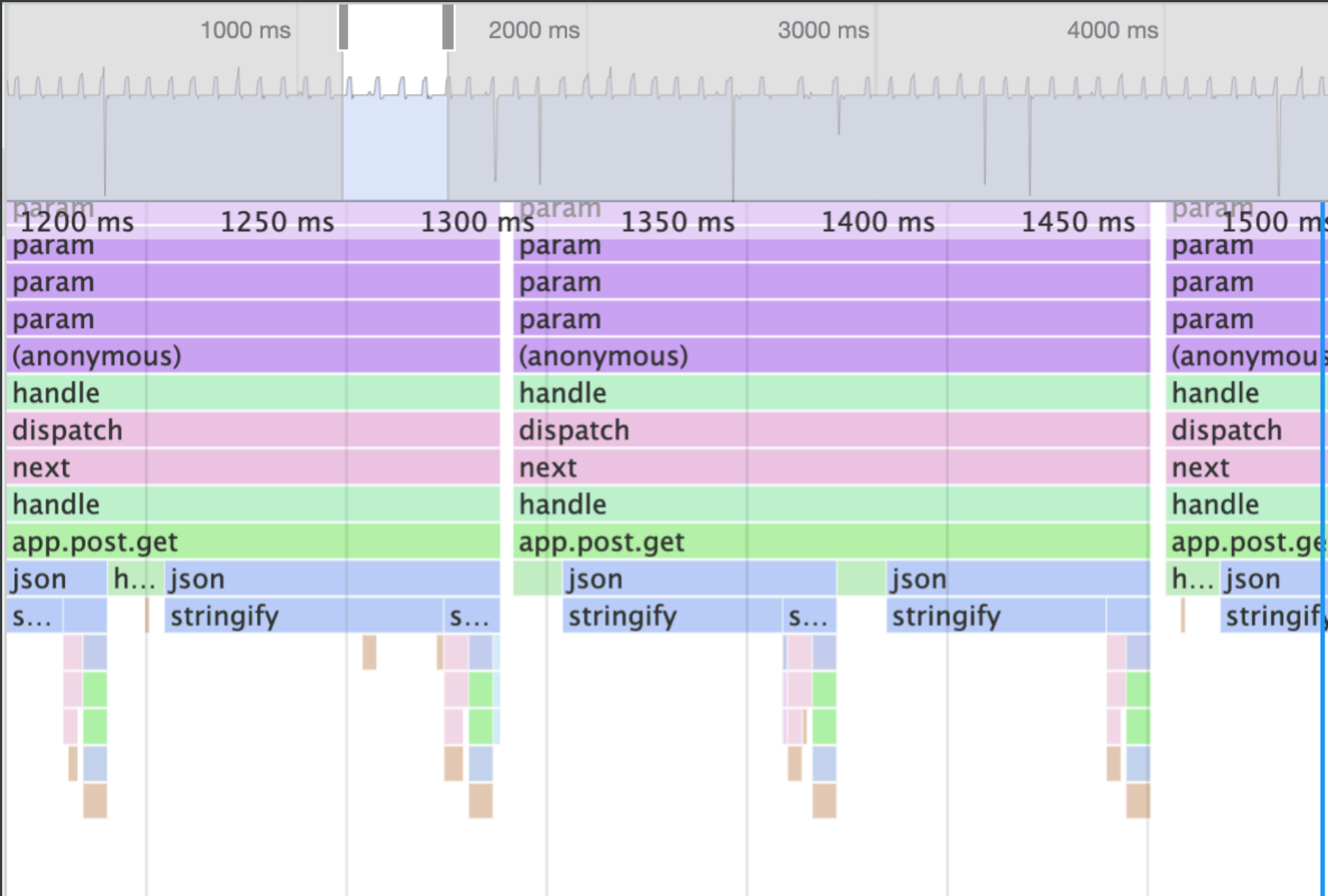
🔥 single-command flamegraph profiling 🔥

Discover the bottlenecks and hot paths in your code, with flamegraphs.

Measure Performance Under Load

Determine where bottlenecks are in your application.

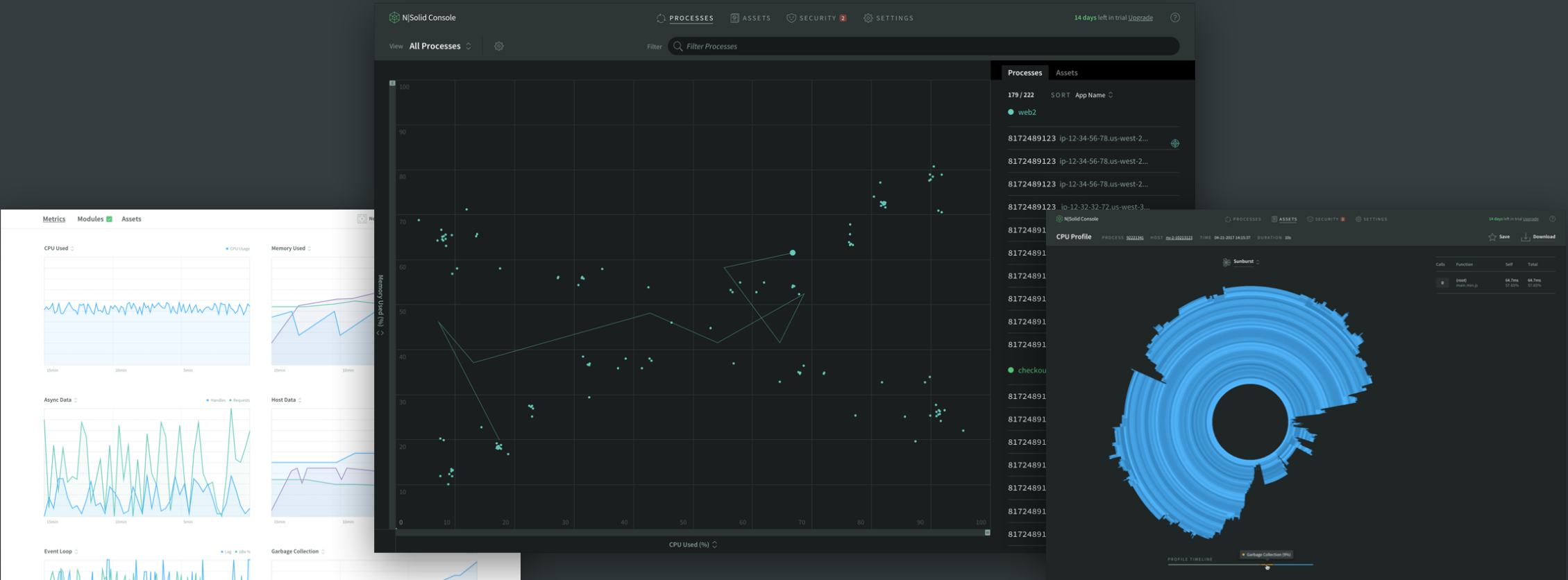
Chrome Devtools



N|Solid

Measure Performance Under Load

Determine where bottlenecks are in your application.



Deoptigate

Understand Optimizations/Deoptimizations

Determine if and why V8 runs your code slower than expected.

FILESDETAILS|Highlight CodeLow Severities

126this.saxParser.onend = (function(_this) {
127return function() {
128if (!_this.saxParser.ended) {
129this.saxParser.ended = true;
130return _this.emit("end", _this.resultOb
131}
132};
133})(this);
134this.saxParser.ended = false;
135this.EXPLICIT_CHARKEY = this.options.exp
136this.resultObject = null;
137stack = [];
138attrkey = this.options.attrkey;
139charkey = this.options.charkey;
140this.saxParser.onopentag = (function(_this
141return function(node) {
142var key, newValue, obj, processedKey, ref
143obj = {};
144obj[charkey] = "";
145if (!_this.options.ignoreAttrs) {
146ref = node.attributes;
147for (key in ref) {
148if (!hasProp.call(ref, key)) continue
149if (!(attrkey in obj) && !_this.optio
150obj[attrkey] = {};
151}
152newValue = _this.options.attrValuePro
153processedKey = _this.options.attrName
154if (_this.options.mergeAttrs) {
155this.assignOrPush(obj, processedKe
156}
157else {
158obj[attrkey][processedKey] = newV

OPTIMIZATIONSDEOPTIMIZATIONSINCLINE CACHES

157 <anonymous> at ./xml2js/lib/parser.js:145:22

Timestamp	Bailout	Reason	Inlined
546ms	eager	wrong map	no
631ms	eager	wrong map	no
721ms	eager	wrong map	no

156 <anonymous> at ./xml2js/lib/parser.js:183:20

Timestamp	Bailout	Reason	Inlined
356ms	eager	out of bounds	no

158 <anonymous> at ./xml2js/lib/parser.js:271:23

Timestamp	Bailout	Reason	Inlined
546ms	eager	wrong map	no
631ms	eager	wrong map	no
721ms	eager	wrong map	no

Demo



h

Getting Setup

- Try N|Solid free after signing up for a free account
 - <https://nodesource.com/products/nsolid>
- Install deoptigate ``npm install -g deoptigate``
 - <https://github.com/thlorenz/deoptigate>



DEVELOPER

Free!

For individual developers and open-source practitioners with Node.js and JavaScript projects they want to monitor.

Resources V8

- **V8-perf resources**
- V8 compiler
- Language Features
- V8 code search

Resources Blog Posts And Documentation

- [deoptigate](#)
- [N|Solid Overview and Documentation](#)
- [Why the New V8 is so Damn Fast](#)

Thank you.

Thorsten Lorenz

@thlorenz