Deno

Key Features

Code Example

Conclusion

# Deno

● Andranik Barseghyan ●

2020

https://github.com/Rebiss/Presentation

# Deno

## A secure runtime for JavaScript and TypeScript.

**Rust.**

Deno is written in Rust.

**V8.**

The code is executed using the same V8.

**Tokio.**

Tokio is used as an Event loop in Deno, again written in Rust

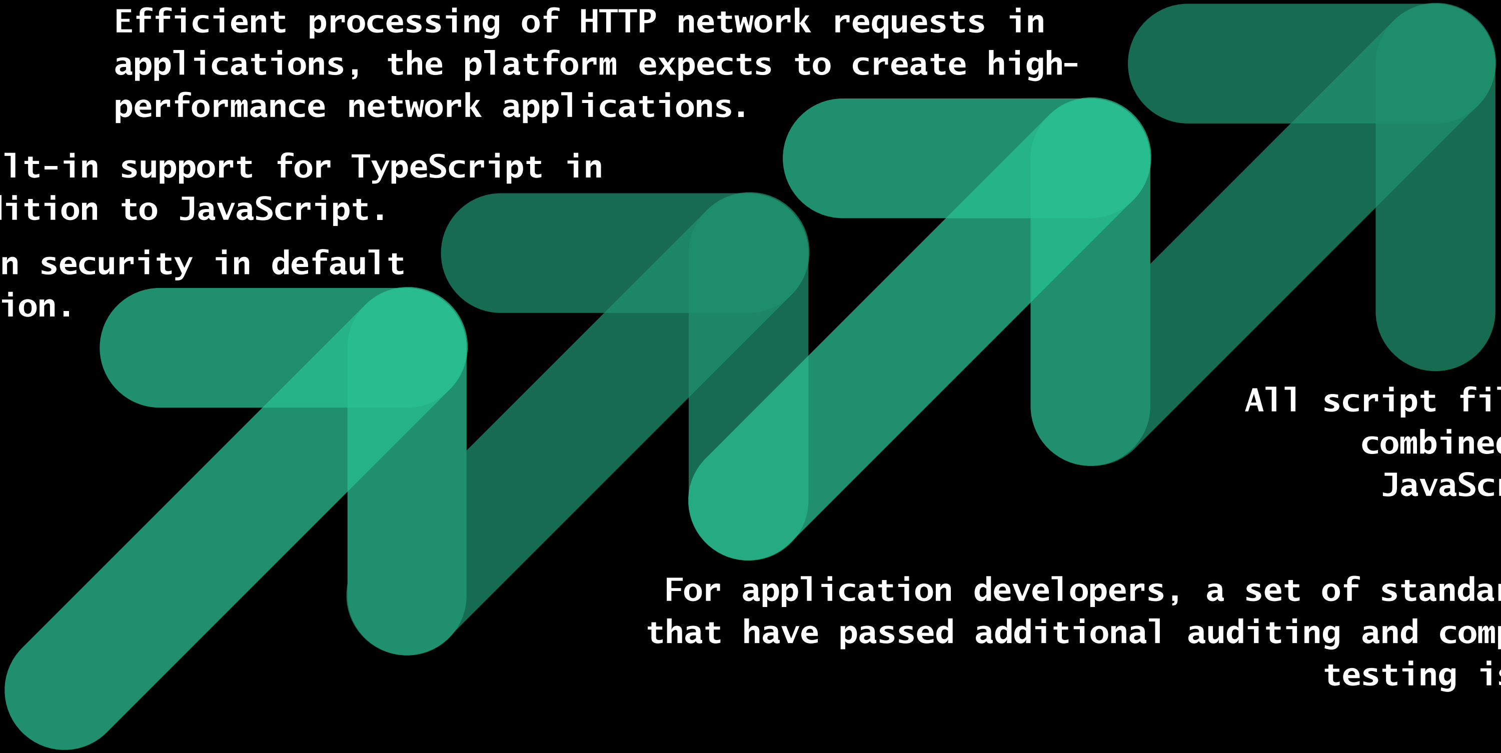
# Key features

In addition to the runtime, the Deno platform also acts as a package manager and allows you to access modules within the code by URL.

Efficient processing of HTTP network requests in applications, the platform expects to create high-performance network applications.

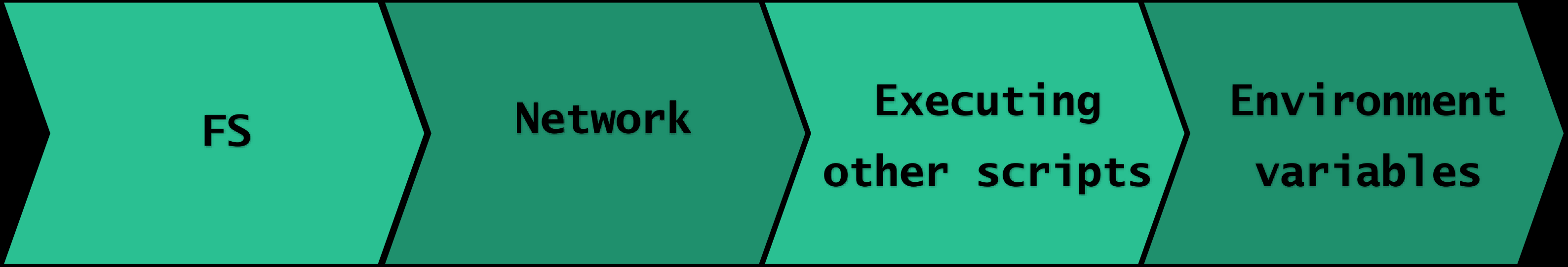
Built-in support for TypeScript in addition to JavaScript.

Focusing on security in default configuration.



All script files can be combined into one JavaScript file.

For application developers, a set of standard modules that have passed additional auditing and compatibility testing is offered;



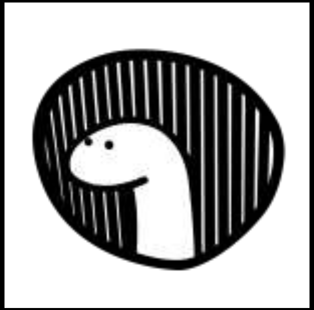
FS

Network

Executing other scripts

Environment variables

Deno requires explicitly defining all necessary permissions for files, network, and environment variables;



# Difference

Does not use NPM,
package.json

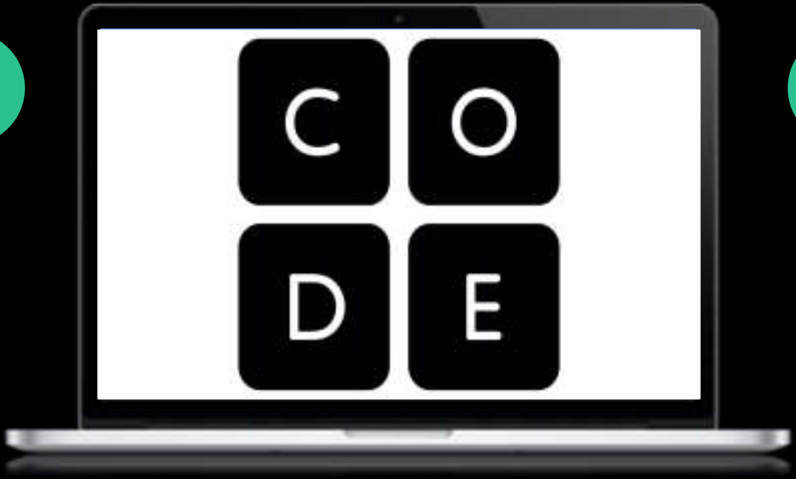
API difference, all asynchronous actions in Deno return a Promise

Deno uses ECMAScript module system and does not support require ()

# Code Example

WebSockets

Chat Example



Run Code

Basic Example

https://github.com/Rebiss/Deno

Problems

Perspective

# Conclusion

Integration

# THANK YOU

## Source

- ✓ https://deno.land/
- ✓ https://deno.land/manual
- ✓ https://github.com/denoland/deno
- ✓ https://twitter.com/deno_land