Node.js in 2021

New node.js features (14.x and 15.x)

- Multithreading for long blocking and scale
- QUIC or HTTP/3 for mobile and 5G
- WASI, N-API 7 in Node.js v15
- V8 v8.6 in Node.js v15
- API new & unknown: diagnostic API, perf_hooks, timer promises, AbortController, vm, v8, fs.watch, crypto.scrypt, async_hooks, AsyncLocalStorage

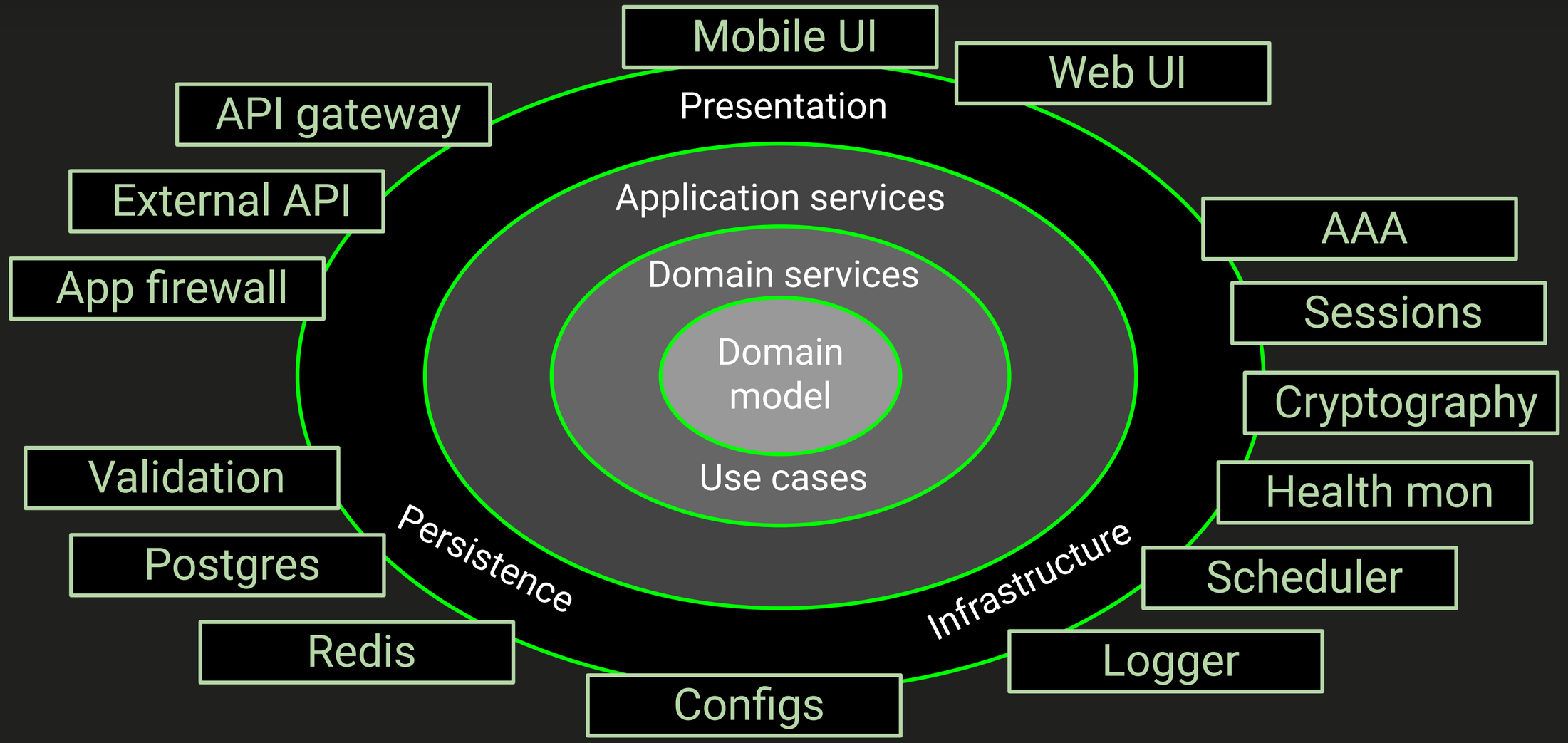
Node.js challenges for next decade

- Enterprise software requirements: maintainability, reliability, high availability, security, isolation, compatibility, performance
- Community culture: meetups, conferences, fundamental knowledge share, consensus
- Reputation and trust: education, certification, community, engineering culture, success stories

Key points of engineering culture

- Architecture: DDD and clean architecture, layered/onion architecture, domain in the middle
- Principles and patterns: GRASP, SOLID, GoF, IoC, DI, coupling and cohesion, Law of Demeter
- Project structure: don’t depend on frameworks
- Techniques: error handling, asynchronous and parallel programming

Application layered (onion) Architecture



New node.js features (14.x and 15.x)

- Multithreading for long blocking and scale
- QUIC or HTTP/3 for mobile and 5G
- WASI, N-API 7 in Node.js v15
- V8 v8.6 in Node.js v15, ES modules (import mjs)
- API new & unknown: diagnostic API, perf_hooks, timer promises, AbortController, vm, v8, fs.watch, crypto.scrypt, async_hooks, AsyncLocalStorage

New strategy for Node.js

- Application is a place only for for applied code
- Use threads to scale, one port per thread
- Stateful applications: performance, interactivity
- Use graceful shutdown and app live reload
- Context isolation and sandboxing for security
- Dependencies (read, be ready to support)
- Configuration is a code

New strategy for Node.js

- No long blocking operations
- Prevent memory and resource leaks
- Remember about race conditions
- Serve static from memory or cdn
- Continuous integration, deployment
- IoC practices and DI (dependency injection)
- No mutable global and prototype pollution

New strategy for Node.js

- No mixins and reference pollution
- Layers instead of middlewares
- Limit async concurrency, queue and timeouts
- Automatic routing instead of manual
- SQL and query builder instead of ORMs
- Error handling instead of error ignoring
- RPC over sockets instead of CRUD over REST

github.com/metarhia

github.com/tshemsedinov
youtube.com/TimurShemsedinov
github.com/HowProgrammingWorks/Index

Largest Node.js and JavaScript course
(186 lectures) https://habr.com/ru/post/485294/

t.me/HowProgrammingWorks
t.me/NodeUA

timur.shemsedinov@gmail.com