Ensuring Your Server-side App Performs

Peter Kellner
DEVELOPER, CONSULTANT AND AUTHOR

@pkellner linkedin.com/in/peterkellner99 PeterKellner.net

Next.js Performance Wins out of the Box

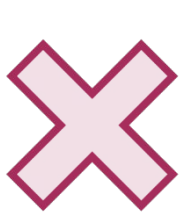
- Code splitting by default
- Prefetching pages async
- ETags automatically generated
- Optimized for React Babel config
- Async script tag loading by default
- Server-side rendering just works

Performances Improvements Coming Up

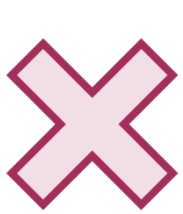
- Better image handling
- Cache React pages inside our Node server
- Setup CDN for Code Split JavaScript and images
- Implement placeholder images while waiting for data to download

Are there downsides to server-side rendering?

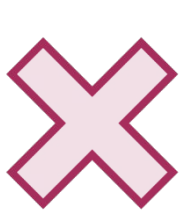
Downsides of Server-side Rendering



Every page landing causes a complex Node invocation



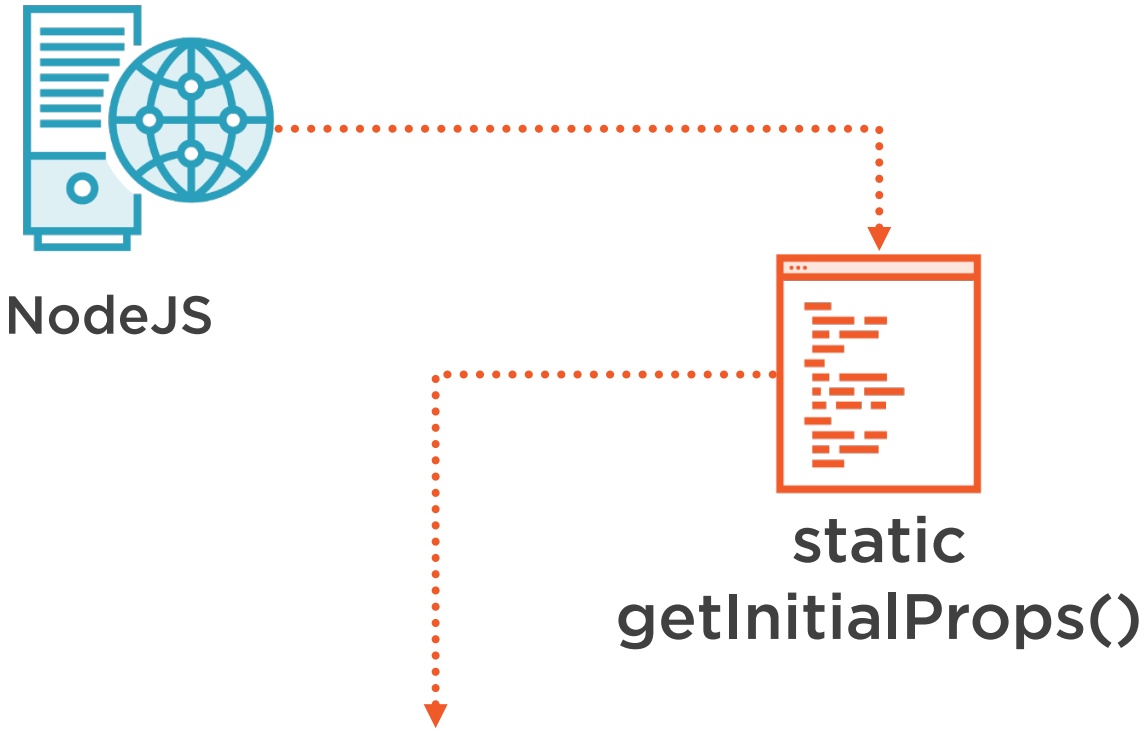
High volume sites typically have high page repetition



High volumes sites must implement caching schemes

CDN: Content Delivery Network

First Page Load from Node Server



Single Responsibility Principle

"Gather together the things that change for the same reasons. Separate those things that change for different reasons."

Martin Fowler

Takeaways

Some great tips on web performance

Same principles as always

Next.js makes implementations easy