The most beautiful thing to happen to UIs today is componentization. It just makes us more efficient, and our projects more manageable. It has also made way for new tooling that has brought even more efficiency. Browser-based tools like the react inspector, for example, are also able to analize large and even complex UIs down to their little components.

Math would put this as structure = efficiency. Interestingly, to attain more efficiency, we know exactly where to touch - the left side of the equation. And this is the opportunity we set out to explore with chtml.

CHTML is a componentization framework for HTML. It is aptly named as it shifts the entire idea of components from JavaScript to plain HTML. And this is an important change!.

Universality with environments and tools.
Web crawlers, universal server-side rendering, more room for tooling if no overengineering.
HTML is a universal language used in both server and client environments. But modern UI components have not been as universal, as these have been largely implemented in JavaScript. For example, server-side rendering with these approach requires a node js environment.

A more universal componentization language that's as universal as HTML itself.
Also, with HTML beibg the webs universal UI markup language, creating UIs should be possible for every web designer. Unfortunately making these UIs has now almost completely shifted from design to core programming.

Defining components in HTML is the answer to making our components work everywhere HTML works, and with everyone that works with HTML.

Interestingly, a simple naming convention does the job perfectly.

This is just conventions and not predefined names.

Heavy tooling to subsidized the high level of complety altogether over engineering

every components and the UI at large down to clear identifiable

Html is about form and function and is best expressed in components. But styling is a work of art and must be approached with more fine grained composability not components. It is on this understanding that we base the design and architecture of access.

in the spirit of art. And this is the important difference css has made with fine-grained, comparable classes for everything art requires.

With approach a, when u create a component, you must also

So theres only a few days left before 2020... I thought you would already be coding by Jan.

Many developers can no longer remember the days of server-side includes.

We missed how plain html made us feel. UI componentization seems to have come at the cost of burying markup inside JS - sinking presentational semantics into scripts - something we once frowned at. The worst part? The .html file extension is fast going extinct as more UI frameworks now encourage baking logic, markup and even styling altogether in JS.

Lest the community has forgotten so fast, many books still in the libraries

The ratio of coding to markup

It takes only about an hour to be sane again on the UI. Get it right

We wanted to break one rule to heed all other rules... instead if the other way around. Proxy... new browsers only.

But u write future-proof code.... the future were heading towards is one with perfect proxies. So going forward spells more irrelevance to hacky work arounds.

The web has it now. Let's dump the work arounds. Write futureproof code that gains more relevance going forward, not less relevance. Dont get caught up in the hacks. The hacks you see today, wont be there tomorrow.

Venia Hub (www.veniabusinesshub.com)

Leadspace Shared Facilities Limited (www.theleadspace.co)

ReDahlia Workspaces (www.redahliaworkspaces.com)

Universal Voucher (www.lagosinnovates.ng)

Capital Square Workspace Solutions Ltd (www.capitalsqua.re)

Workstation (www.workstationng.com)

Cranium One (www.cranium-one.com)

Seedspace Lagos (https://www.seedspace.co/lagos-nigeria/seedspace-lagos/)

Cantagali Co-working Space (www.cantagali.com)

Exponential Hub (https://www.simplyexponential.com/hub)

The Village (http://village.ng)

Impact Hub (http://impacthub.net)

Last modified: 8 Dec 2019