Paginating or infinite scrolling web content?

The web has always been a scrolling medium but there are reasons and motivations that will make people decide for one or the other. There is no perfect 'one-size-fits-all' solution and you will have to evaluate which solution works best for your project. In this post we'll discuss advantages and disadvantages of pagination and scrolling and suggest one combined solution that may work well as a generic solution when working outside of frameworks.

In Pagination vs. Scrolling: The Great Website Debate ThriveHive presents advantages and disadvantages of both pagination and scrolling for web sites. They come to the same conclusions most developers have: it depends.

But just like the Nielsen Norman Group reminds us that Infinite Scrolling Is Not for Every Website the same case can be made for pagination.

Scrolling

Scrolling, in this context, refers both to the regular scrolling of a web page and to infinite scrolling: a technique where the browser loads content as the user reaches the bottom of the screen so it appears as if the content will scroll continually as long as there is new content available to display.

Vue, React and Angular

All the frameworks I've reviewed have some sort of virtual/infinite scrolling tool available. These are the ones I found, I'm pretty sure there are more:

- React Virtualized in general and the List component in particular
- Angular CDK's `scrolling` package
- Vue virtual scroller

Web Platform

Surprisingly the web platform doesn't provide built-in mechanisms for infinite scrolling and there several items necessary for successful infinite scrolling that are not part of the web platform yet.

One,incomplete, example is this pen from Werner Beroux that uses vanilla JavaScript to generate an infinite scrolling list.

At the 2018 Chrome Dev Summit, Gray Norton presenteda new tool called the virtual scroller; a set of custom elements that will do the heavy work of creating virtual scrolling for you.

The Github repository has two different concepts for virtual scrolling. Check the project's readme for more information about the branches and what they accomplish.

Pagination

Pagination takes a different approach. Rather than provide infinite scrolling, it breaks the content into "pages" and provides means to navigate the content.

According to the Interaction Design Institute:

> Pagination is the process of splitting the contents of a website, or a section of contents from a website, into discrete pages. This user interface design pattern is what we designers use to save site visitors from being overwhelmed by a mass of data on one page – we take that 'continental' chunk and splinter it sensibly into 'islands', literally distinct pages which users will be able to devote their attention to without sighing in

> exasperation.

Depending on the type of content that you're working with, pagination may or may not be the best solution. In 2013 Jakob Nielsen warned us that "listings might need pagination by default, but if users customize the display to View All list items, respect that preference."

Working with long-form content works differently. We definitely want to paginate books or long essays... but we still need to be mindful of how we organize the pagination for this type of content.

Frameworks

Someone has thought about pagination for the frameworks I look at regularly so there shouldn't be a problem with implementing the pattern in any of these frameworks.

- Creating a Reusable Pagination Component in Vue
  - Example in codesandbox
- React paginate
- NGX Pagination

Web Platform

A possibility is to use CSS scroll snap as a way to navigate between sections of content.

An example of scroll snap shows how it works with mouse events. A next step would be to convert the mouse events into pointer events to make sure we cover both dekstop and mobile devices.

Something worth researching is whether it's possible to copy the pagination tools from frameworks like Foundation or Bootstrap 4 without having to use the whole framework.

So which one do we use

As with many things dealing with the web, it depends. It depends on the type and quantity of material that we want to display and what our target devices are.

For book-like content it would be ideal to paginate the content at the chapter level and, inside the chapter, let the content scroll as necessary. This will give us the best of both worlds in situations where physical books would just add pages.