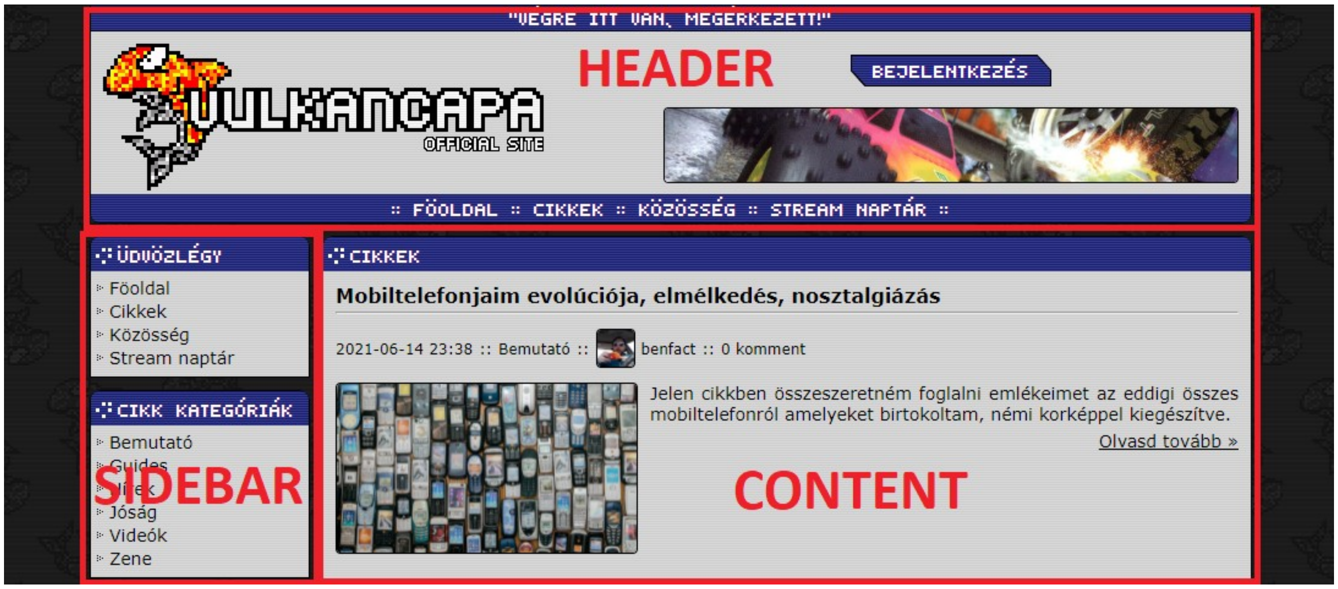
An introduction into React

What is React?
React is a JavaScript library created by Facebook. React is a declarative, efficient, and flexible JavaScript library for building user interfaces. It lets you compose complex UIs from small and isolated pieces of code called “components”.

The best way to make interactive websites is by building it up from components! This is why is a smart choice to use React.

Why build a website from components?



Mobiltelefonjaim evolúciója, elmélkedés, nosztalgiázás

2021-06-14 23:38 :: Bemutató :: benfact :: 0 komment

Jelen cikkben összeszeretném foglalni emlékeimet az eddigi összes mobiltelefonról amelyeket birtokoltam, némi korképpel kiegészítve.

Here’s an example. This is a website that has 3 main elements. A *Header*, *Sidebar* and a *Content* block. If you navigate on the site, only the *Content* region changes, everything else remains the same. The basic structure of a site like this, is having a file, where you include the header component, sidebar component and the content you want to display on the page (or this can be a component too, depends on what you want to do).

This is a very smart way to build a website, because if you have, let’s say... about 20 pages, where you included one component, and you want to change it, you only have to edit 1 file, not all 20 pages. It’s also important here to not have the same code over and over written in the files, if you have something that appears more than once on a page, it should be an includable component! This will help you and others to have a much readable, cleaner and smaller code.

How does React work?
React creates a VIRTUAL DOM in memory. Instead of manipulating the browser's DOM directly, React creates a virtual DOM in memory, where it does all the necessary manipulating, before making the changes in the browser DOM. React only changes what needs to be changed! React finds out what changes have been made, and changes only what needs to be changed. This is why the most people choose this library to make single-page-applications!

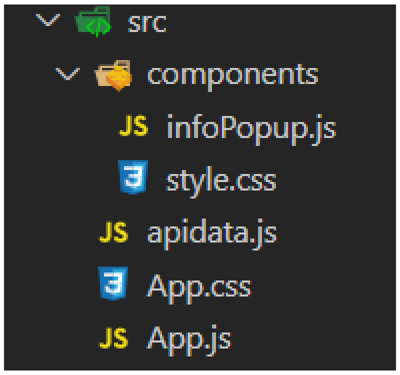
Sidenote:
Why it's *soooooooo* important to choose the right library and framework to your project?
Every library and framework mostly made for a few, or one purpose. When you start a project for a customer, you have to know very detailed what he wants to see. This is why some excellent tools are exists, like Figma, to make quickly webpage layouts and designs, and frameworks and library to quickly throw a techdemo about his imagination. It's really important to do some research about the framework or library you want to use, for what use cases will fit the thing you want to use. Example: You are not going to build a social site from Wordpress. It's made for simple blogs, not else. Also be careful to not over-engineer your code. Here's a great video about that: https://www.youtube.com/watch?v=-AQfQFcXac8
Also, think about the future. What if the client wants a calendar where he add some events. That's cool. You found a very simple library for it, you completed the task in 30 minutes. But what if he wants next week drag 'n drop features, pictures, and mouse hover features? Did the library you choose is capable for these? Or you have to hack somehow into the library because you can't make these easily. If you want to choose this way, I recommend you to just write it from the ground in the way you need, trust me, you will thank me later this tip. The last thing you have to follow, is to not have many unused things in your project. Example: You want an arrow icon, just an arrow icon, nothing else. You include the WHOLE FontAwesome library in your project just for one svg file. Don't do this, or you will end up like this:

When you want to make a website and you use all the cool sounding libraries/-frameworks you ever heard of:
What you imagined:
What you got:
jQuery
npm
2.1 MB resources | Finish: 443 ms
9.9 MB resources | Finish: 6.19 s

Sample Project

For a small demonstration of React, I made a simple single page application that uses an API. Here's where you can download it: https://github.com/stephenmiller04/reacttutorial

The application is really simple, you only need to check out a few files to know how it works:



```
src
  components
    infoPopup.js
    style.css
  apidata.js
  App.css
  App.js
```

These are the files I worked in it. The main file where the bigger part of the site are stored (and the logic) if the App.js file. You should open it, because I commented everything in it you need to know!

In the App.js I declared every function I needed for the site, and all the variables I use. There's a function called **getMovies()** where is the logic to themoviedb.org's API. In the header I included the apidata.js file. It's recommended to store these variables in a separate file for better code structure and reuse for later. Below this function there's four other that calls the API with the correct variables, they are simple page stepper functions.

The last function in this file is the **render()**. This is where you build your page you want to see when you open your project. In this example there are 3 main div's here. On the top there are the search bar, below you can find the navigation bar, and after the movie cards.

There's an example in this sample for a component I talked about earlier. It's a popup where you click on the movie you want more information about and it will displays on the site in full screen. It's called **infoPopup.** In the render() there's a div with the class name "movie-card" where you can see where it's called. Also the component is declared at the start of the render() function, and you can see what variables it uses.

That's for this documentation, now check out the sample project and modify it! I tried to comment every important thing in there.