[http://localhost:5000/pixabay/index.html](http://localhost:5000/exercises/1/index.html)

<https://pixabay.com/api/docs/>

**task:**

By the way sweetie, when you are ready to learn about HTML, start reading about it and then do the exercise here: <https://frontend.2020.plus.jakobkallin.com/1/>

[15:45, 23/06/2020] Jakob: To write HTML for your CV

[15:45, 23/06/2020] Jakob: Then I will use that HTML for my CSS.

[16:58, 24/06/2020] Jakob: Well, write HTML for everything that you want on your webpage

[16:58, 24/06/2020] Jakob: After that, we will make sure we can add the CSS to your HTML.

[16:58, 24/06/2020] Jakob: You don't need any CSS yet

[16:58, 24/06/2020] Jakob: The default CSS in the browser means your page will look OK as long as you select good HTML elements

------------------------.

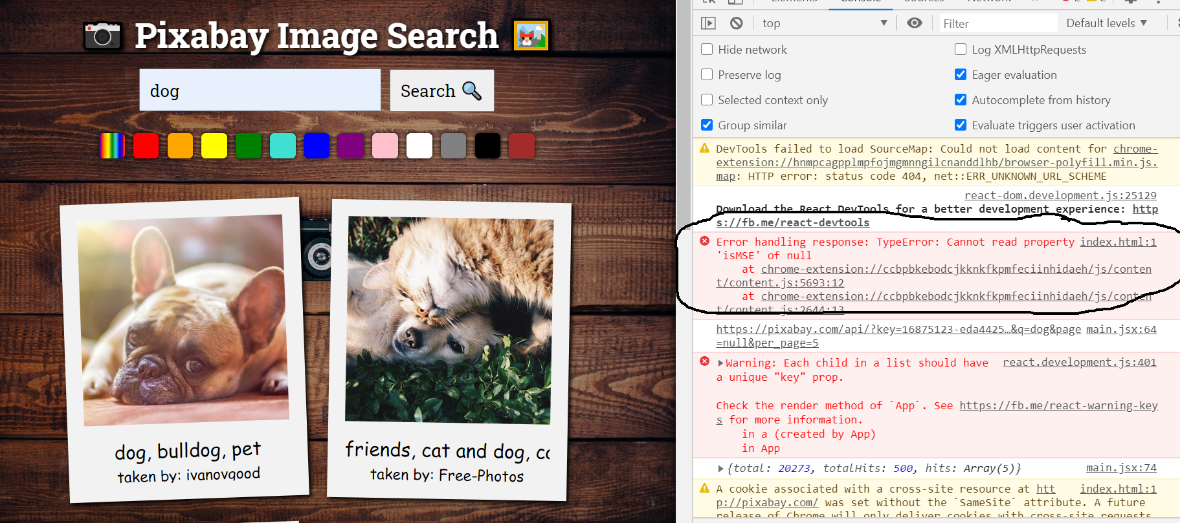
=================================\*

**HTML to JSX Compiler**

<https://magic.reactjs.net/htmltojsx.htm>

If you want to call both fetchSearchResult and setState in the same method, you should do it separately

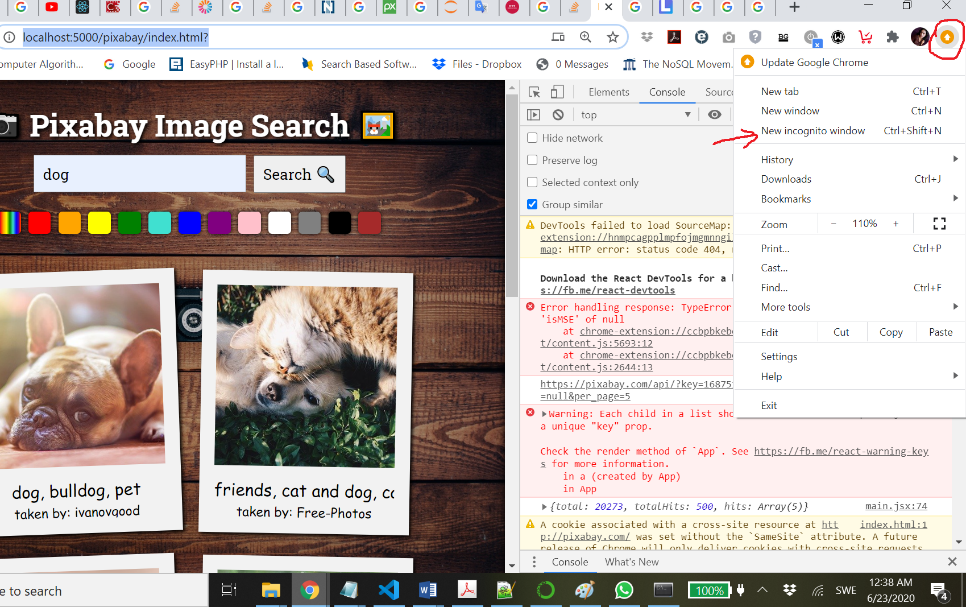
Right now you are trying to call fetchSearchResult inside setState which is not really possible in this way.



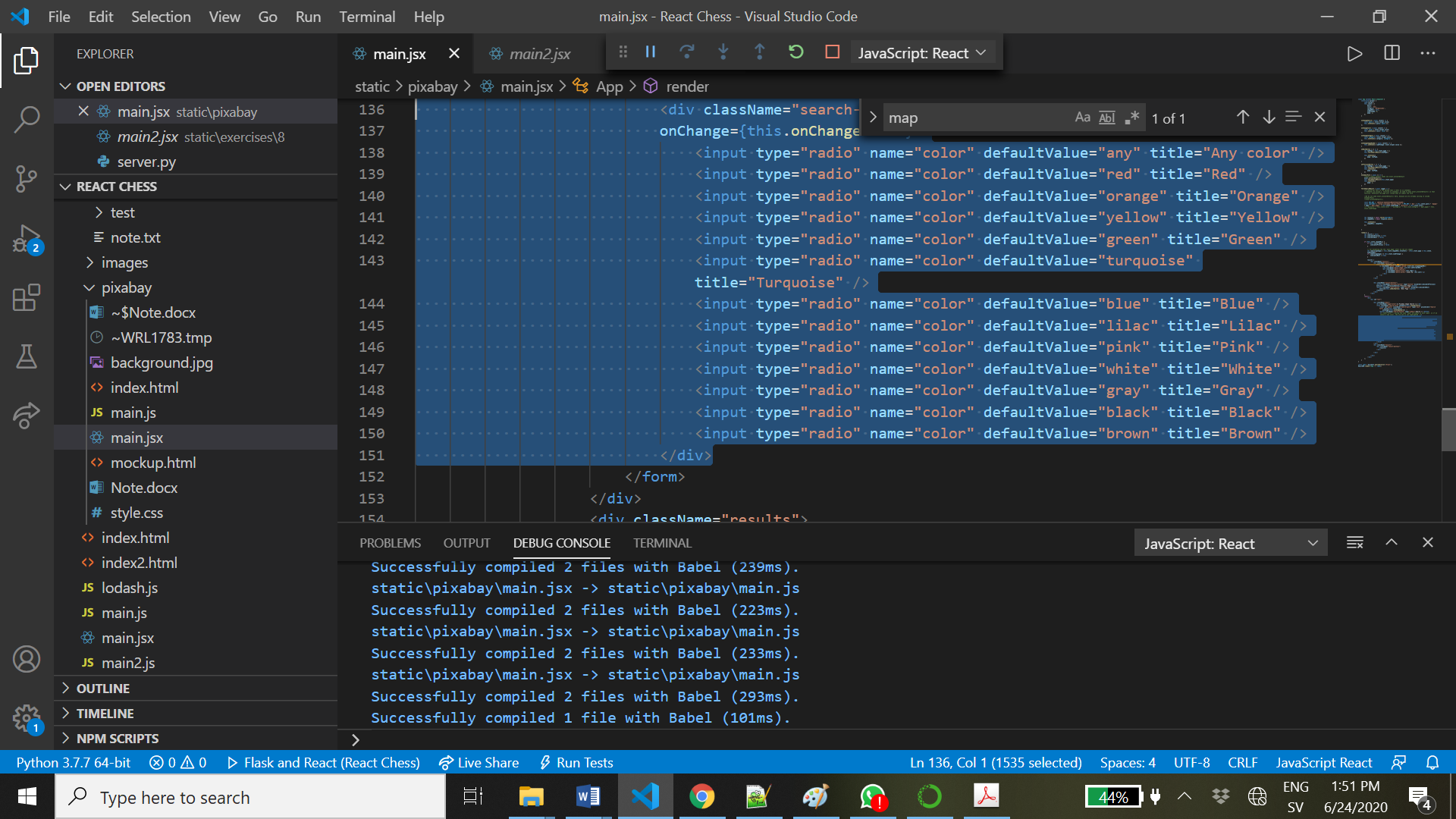
I think it might be one of your extensions

Try it in incognito mode

If it does not appear in incognito, it is one of your extension, because they are disabled in incognito.



if they are weird errors, it's often because of extensions.



You can use a loop or map to avoid repetition.

If you want to use a for loop, you will have to put it before the "return", just like you have done for "results"

With map you can put it in the same place as it is right now (in return).

* Ctrl+D to select the next occurrence of the word you have selected, or Ctrl+Alt+Down to get a second cursor on the line below
* Let's say I want to change "photo" to "image" in this part of the HTML. I can select "photo" on the first line, then press Ctrl+D five times to select the other instances as well. Then I can type in all places at the same time



how about if there is photo variable inside method?

You can use Ctrl+D for that, but then you might also accidentally change something that is not actually the same variable but just happens to contain the string "photo". In VS Code you can instead select the variable and then press F2, to rename it. It often works, but not always, because JavaScript is a dynamic language. In a language like Java or C#, it will always work.

In eclipse there were one way that automatically select all the variable with same name. I meen it was no need to press Ctrl+D five times to choose it:

[14:12, 24/06/2020] Jakob: Yes, in VS Code it's F2

[14:12, 24/06/2020] Jakob: Also in Visual Studio

[14:12, 24/06/2020] Jakob: You can use it for variable names, method names, and some other things.

-----\*

[19:48, 24/06/2020] Jakob: If you filter with "cat" for example, you will only see your searches

[19:48, 24/06/2020] Jakob: Or with "/api"

\*

**Bug:**

Actually this will apply to the search term as well. If we search for cat, then change the text to dog and press Next Button (without pressing Search again), what will happen?

Our state is not able to distinguish between the term that we **actually searched for** and the term that we have **currently typed in the box**

There are actually different ways to fix this based on what functionality you want

Still related to state though

For the page, we need to reset it when we search, as you have done

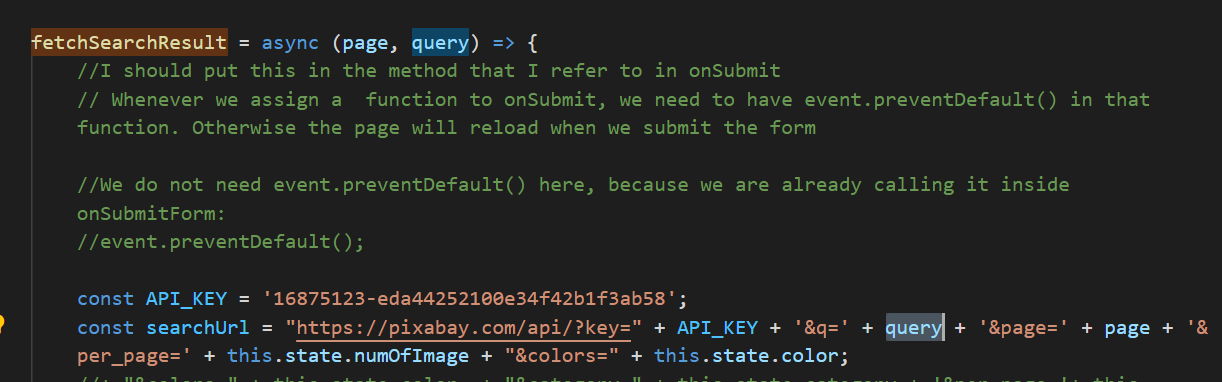
For color and query, there are actually two different things we need to keep track of:

- what value has the user **submitted before**?

- what value has the user **currently entered**?

When we press submit, we change submittedQuery in setState. But setState doesn't actually change this.state until after the function has completed, so this.state.submittedQuery will still be the **old** value

The easiest way to fix this is to **not use this.state.submittedQuery or this.state.currentQuery** in fetchSearchResult. … instead use query which is in parameter



Instead, we should add another parameter to fetchSearchResult

The parameter should be the query to search for, just like we added the page to search for as a parameter

Then we can pass in submittedQuery when we need to and currentQuery when we need to.

**submittedQuery** when we want the **old** one, **currentQuery** when we want the **new** one.

Both submittedQuery and currentQuery must be declared in this.state. Because we need to be able to remember both of them. One of them until we press **Submit**, and the other until we press **Next** Page or **Previous** Page

Is it correct that we need to remember submittedQuery until we press **Submit**, and we need currentQuery until we press **Next** Page or **Previous** Page?

[20:54, 25/06/2020] Jakob: You are setting the state. But remember that the state doesn't change until after the function has completed

[20:54, 25/06/2020] Jakob: And even if it changed immediately, you are changing it after calling fetchSearchResult

**---------------**

**Now let's fix the key thing as well**

Have you noticed that often there is a strange delay when loading new images?

The title of the new images are loaded quickly, but the images are slower, so for a while you actually see the old image with the new name.

It is because the titles load faster than the images, because they are just text

We cannot avoid this, but we can make sure we at least don't see the wrong title with the wrong image.

We can make it so that the image simply is not shown until it is loaded, but the title is shown immediately under a blank image

This is actually the default behavior, but because React does some things to make the GUI quicker and because we are not using "key", we do not get that behaviour.

React uses "key" to know if something has changed or not. If a part of the GUI has a different key, React knows that it is a new part of the GUI and will throw away the old one. If there are no keys, it will actually keep the old one, which is why we are seeing the old image until the new one has loaded.

A key is a value that should be completely unique for that part of the GUI. For example, each of our images should have a key that is the ID of that photo on Pixabay (we can get that from the API). Then we know that when we show a new page of images, React will throw away the old ones because the new ones have different keys.

This is also the reason why we **often should not** use the **index** for our key. React says this in their documentation and error messages. If we use the index as the key, the first image on the page will have index 0, and when we press Next Page, the first image on the page will still have index 0, even though it's a different image

Instead, we want to use a unique ID for that image, which Pixabay has already created for us

=================================\*

We need to use the checked property on radio buttons

And for this it will be easier to have map or a loop to create them, instead of repeating the HTML

Let's see if we can find an example online

https://www.pluralsight.com/guides/how-to-use-radio-buttons-in-reactjs

Look at the last code sample here.



He uses a boolean expression in the "checked" attribute

Comparing to a value in the state

But he doesn't do it in a loop. You can do it in a loop if you create an array of colors

You don't need to add an onChange for every radio. It already works the way we have it now, with just one onChange on the <div>

It's going to be nicer with a loop or map.

If you use **{color}**, you get the value of the variable color: red, blue, green etc.

If you use just value=**"color"**, it will just be the string "color"

If you write this:

this.state.currentColor === "color"

then you will be comparing with the literal string "color". So "red" will be compared to "color", "blue" will be compared to "color" etc.

But if you write this:

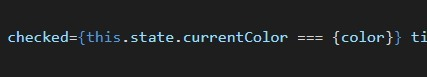
this.state.currentColor === color

then you will be comparing with **the value of the variable color,** which in this case will be "red", then "orange", then "yellow" etc. while it is looping through



That is when you want to use the variable color inside an HTML attribute/property

[00:53, 27/06/2020] Jakob: so it's only in JSX



[01:00, 27/06/2020] Jakob: We only use one pair of { } for each HTML attribute

[01:00, 27/06/2020] Jakob: We don't put { inside {

[01:01, 27/06/2020] Jakob: What { actually does in JSX is that it switches from HTML to JavaScript

[01:01, 27/06/2020] Jakob: Anything inside the braces is JavaScript

[01:01, 27/06/2020] Jakob: Anything inside < is HTML

[01:01, 27/06/2020] Jakob: So if we want JavaScript in HTML, we use {

[01:01, 27/06/2020] Jakob: and if we want HTML in JavaScript, we use <

[01:01, 27/06/2020] Jakob: < and { switch between the two languages in the same file

I know the syntax is invalid because we don't put { inside other { in JSX, and there is no reason to