Absolute vs Relative Paths (Article)

You learned about <Link> , you learned about the to  property it uses.

The path you can use in to can be either **absolute** or **relative**.

**Absolute Paths**

By default, if you just enter to="/some-path"  or to="some-path" , that's an **absolute path**.

**Absolute path** means that it's**always appended right after your domain**. Therefore, both syntaxes (with and without leading slash) lead to example.com/some-path .

**Relative Paths**

Sometimes, you might want to create a relative path instead. This is especially useful, if your component is already loaded given a specific path (e.g. posts ) and you then want to append something to that existing path (so that you, for example, get /posts/new ).

If you're on a component loaded via /posts , to="new"  would lead to example.com/new , **NOT** example.com/posts/new .

To change this behavior, you have to find out which path you're on and add the new fragment to that existing path. You can do that with the url  property of props.match :

<Link to={props.match.url + '/new'}>  will lead to example.com/posts/new  when placing this link in a component loaded on /posts . If you'd use the same <Link>  in a component loaded via /all-posts , the link would point to /all-posts/new .

**There's no better or worse way of creating Link paths** - choose the one you need. Sometimes, you want to ensure that you always load the same path, no matter on which path you already are => Use absolute paths in this scenario.

Use relative paths if you want to navigate relative to your existing path.