Use English language when naming classes, variables and functions.

```
const pahtamaNanme = “Soe”  /* Bad */
const firstName          = “Soe”  /* Good */
```

Pick one naming convention and follow it.

It may be PascalCase or camelCase or snake_case.

```
/* Bad */
const pages_count  = 5
const shouldUpdate = true

/* Good */
const pageCount     = 5
const shouldUpdate = true

/* Good as well */
const page_count     = 5
const should_update = true
```

A name must be short, intuitive and descriptive (SID):

Short: a name must not take long to type and, therefore, remember

Intuitive: a name must read naturally, as close to the common speech as possible

Descriptive: a name must reflect what it does/possesses in the most efficient way

```
/* Bad */
const a = 5                              // “a” could mean anything
const isPaginatable = a > 10      // “Paginatable” sounds extremely unnatural
const shouldPaginatize = a > 10 // Made up verbs are so much fun!

/* Good */
const postCount = 5
const hasPagination = postCount > 10
const shouldDisplayPagination = postCount > 10 // alternatively
```

Avoid contractions

```
/* Bad */
const onItmClk = () => {}

/* Good */
const onItemClick = () => {}
```

Avoid context duplication

```
class MenuItem {
  /* Method name duplicates the context (which is "MenuItem") */
  handleMenuItemClick = (event) => { ... }

  /* Reads nicely as `MenuItem.handleClick()` */
  handleClick = (event) => { ... }
}
```

Reflect the expected result

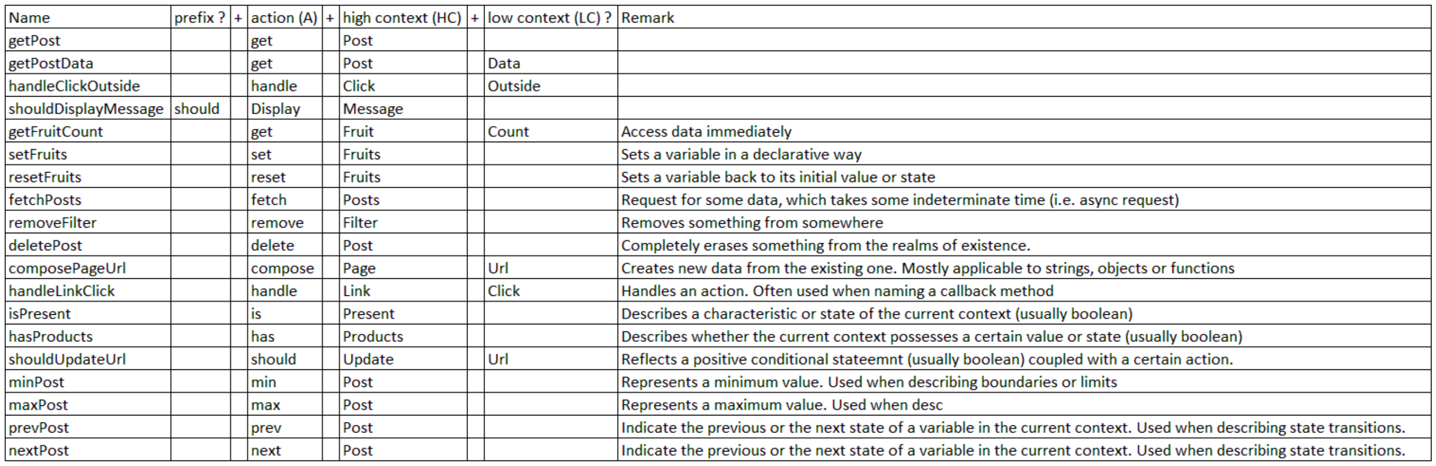
```
/* Bad */
const isEnabled = itemCount > 3
return <Button disabled={!isEnabled} />

/* Good */
const isDisabled = itemCount <= 3
return <Button disabled={isDisabled} />
```

A domain that a function operates on.

A function is often an action on something. It is important to state what is its operable domain, or at least an expected data type.

High context emphasizes the meaning of a variable.

The verb part of your function name. The most important part responsible for describing what the function does.



Name	prefix ?	+	action (A)	+	high context (HC)	+	low context (LC) ?	Remark
getPost			get		Post			
getPostData			get		Post		Data	
handleClickOutside			handle		Click		Outside	
shouldDisplayMessage	should		Display		Message			
getFruitCount			get		Fruit		Count	Access data immediately
setFruits			set		Fruits			Sets a variable in a declarative way
resetFruits			reset		Fruits			Sets a variable back to its initial value or state
fetchPosts			fetch		Posts			Request for some data, which takes some indeterminate time (i.e. async request)
removeFilter			remove		Filter			Removes something from somewhere
deletePost			delete		Post			Completely erases something from the realms of existence.
composePageUrl			compose		Page		Url	Creates new data from the existing one. Mostly applicable to strings, objects or functions
handleLinkClick			handle		Link		Click	Handles an action. Often used when naming a callback method
isPresent			is		Present			Describes a characteristic or state of the current context (usually boolean)
hasProducts			has		Products			Describes whether the current context possesses a certain value or state (usually boolean)
shouldUpdateUrl			should		Update		Url	Reflects a positive conditional stateemnt (usually boolean) coupled with a certain action.
minPost			min		Post			Represents a minimum value. Used when describing boundaries or limits
maxPost			max		Post			Represents a maximum value. Used when desc
prevPost			prev		Post			Indicate the previous or the next state of a variable in the current context. Used when describing state transitions.
nextPost			next		Post			Indicate the previous or the next state of a variable in the current context. Used when describing state transitions.