Redirect user into another page with window.location in javascript

There are three main ways to redirect to another URL with JavaScript:

1. **window.location.href**
2. **location.assign()**
3. **location.replace()**

<u>**Window.location.href**</u>

The use of "window.location.href" for redirection simulates the action of clicking on a hyperlink. It adds a new entry to the browser's session history. This allows users to use the "back" button to return to the previous page.

window.location.href = 'https://exampleURL.com/';

<u>**Window.location.asign( )**</u>

You can use "**window.location.assign**" as an alternative to "**window.location.href**" for redirects.

The syntax is:

```
window.location.assign('https://www.exampleURL.com/');
```

Similarly to "window.location.href," it directs the browser to load the specified URL and add this new page to the session history—allowing users to use the browser's "back" button to return to the previous page.

both "window.location.assign()" and changing the location.href property should work just fine. The choice between them is mostly a matter of personal preference and coding style.

Window.location.replace( )

Like "window.location.assign()", "**window.location.replace()**" calls a function to display a new document at the specified URL.

The syntax is:

```
window.location.replace('https://www.exampleURL.com/');
```

Unlike setting a new location property or using "window.location.assign()" the replace method does not create a new entry in your session history.

Instead, it replaces the current entry. Meaning users cannot click the “back” button and return to the previous, pre-redirect page.

Such as in **Login redirects, User authentication, Form submission success.**

checkValidity() method

The HTMLSelectElement.checkValidity() method checks whether the element has any constraints and whether it satisfies them. It returns **true** if the value of the element has no validity problems; otherwise returns **false**, If the element fails its constraints, the browser fires a cancelable invalid event at the element, and then returns false.

setCustomValidity() method

The setCustomValidity() method of the HTML object element interface sets a custom validity message for the element.

setCustomValidity(errorMessage)

It's vital to set the message to an empty string if there are no errors. As long as the error message is not empty, the form will not pass validation and will not be submitted.