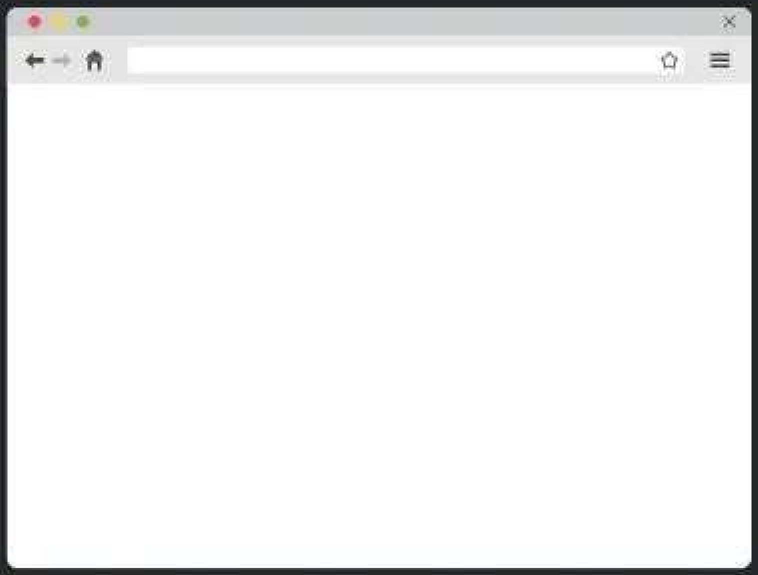
JavaScript Session Storage

@Aakash Kanojiya

Introduction

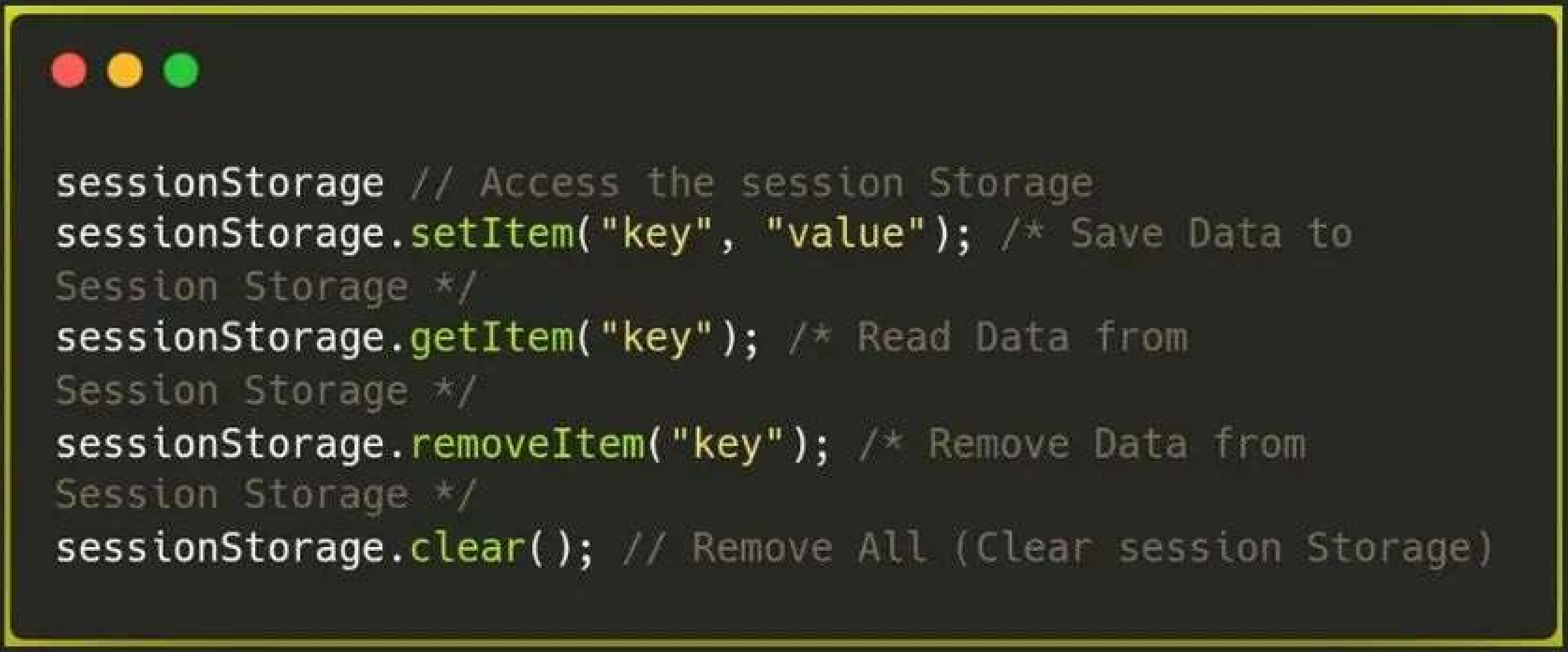
The sessionStorage object let you store key/value pairs in the browser.

It stores data only for a session. It means that the data stored in the sessionStorage will be deleted when the browser is closed.



@Aakash Kanojiya

Managing data in the JavaScript sessionStorage

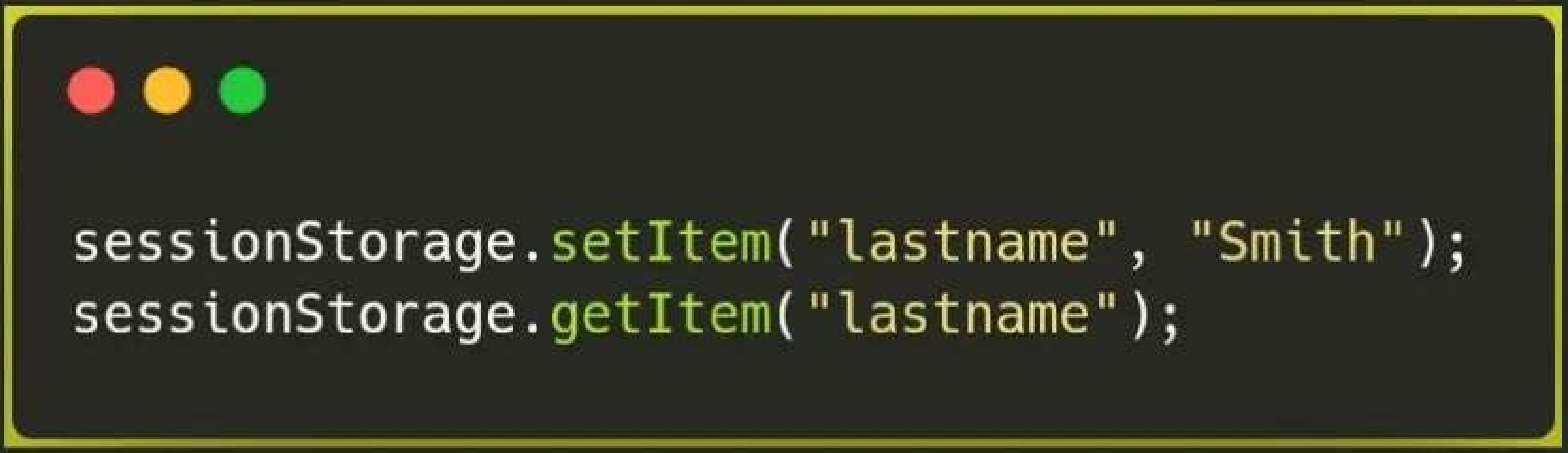


```
sessionStorage // Access the session Storage
sessionStorage.setItem("key", "value"); /* Save Data to
Session Storage */
sessionStorage.getItem("key"); /* Read Data from
Session Storage */
sessionStorage.removeItem("key"); /* Remove Data from
Session Storage */
sessionStorage.clear(); // Remove All (Clear session Storage)
```

@Aakash Kanojiya

Example:

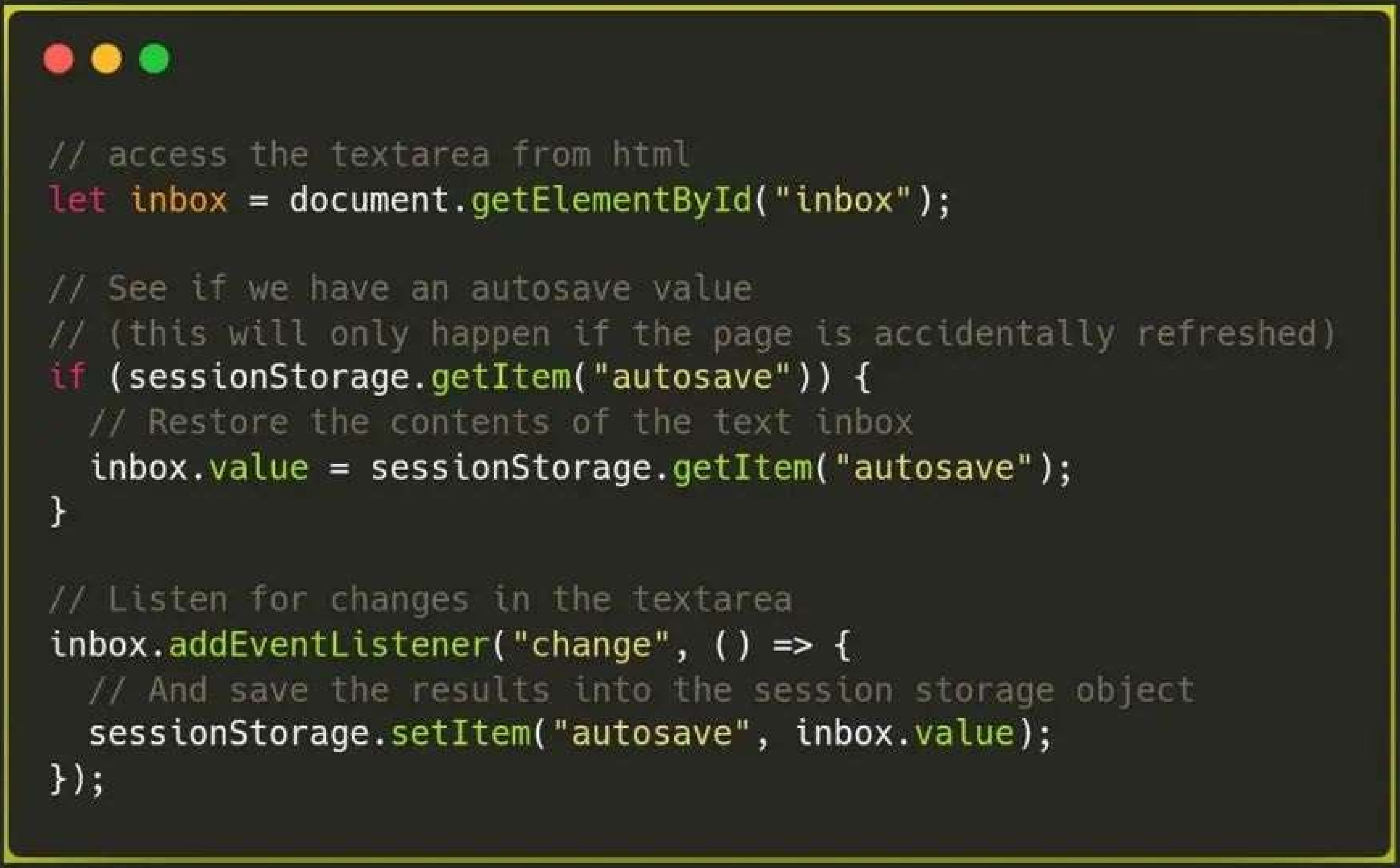
Set and retrieve a sessionStorage name/value pair:



```
sessionStorage.setItem("lastname", "Smith");
sessionStorage.getItem("lastname");
```

@Aakash Kanojiya

Real Example:



```
// access the textarea from html
let inbox = document.getElementById("inbox");

// See if we have an autosave value
// (this will only happen if the page is accidentally refreshed)
if (sessionStorage.getItem("autosave")) {
  // Restore the contents of the text inbox
  inbox.value = sessionStorage.getItem("autosave");
}

// Listen for changes in the textarea
inbox.addEventListener("change", () => {
  // And save the results into the session storage object
  sessionStorage.setItem("autosave", inbox.value);
});
```

The above example autosaves the contents of a textarea, and if the browser is refreshed, it restores the textarea content so that no writing is lost.

Why use sessionStorage?

1. The sessionStorage can be used to store the state of the user interface of the web application. Later, when the user comes back to the page, you can restore the user interface stored in the sessionStorage.
2. The sessionStorage can also be used to pass data between pages instead of using the hidden input fields or URL parameters.

DID YOU FIND IT USEFUL?

FOLLOW FOR MORE

@Aakash Kanojiya