enum ScrollRestoration { "auto", "manual" };

[Exposed=Window]

interface History {

readonly attribute unsigned long length;

attribute ScrollRestoration scrollRestoration;

readonly attribute any state;

undefined go(optional long delta = 0);

undefined back();

undefined forward();

undefined pushState(any data, DOMString unused, optional USVString? url = null);

undefined replaceState(any data, DOMString unused, optional USVString? url = null);

};

Описание свойств:  
history.length✔MDN

Возвращает общее количество записей истории сеансов для текущего проходимого объекта навигации.

history.scrollRestoration✔MDN

Возвращает режим восстановления прокрутки для записи истории активных сеансов.

history.scrollRestoration = value

Установите для режима восстановления прокрутки записи истории активных сеансов значение value.

history.state✔MDN

Возвращает состояние классического API истории записи истории активных сеансов, десериализованное в значение JavaScript.

Описание методов:

history.go()

Перезагружает текущую страницу.

history.go(delta)✔MDN

Возвращает назад или вперед указанное количество шагов в общем списке записей истории сеансов для текущего проходимого элемента навигации.

Нулевое отклонение приведет к перезагрузке текущей страницы.

Если дельта находится вне диапазона, ничего не делает.

history.back()✔MDN

Возвращается на один шаг назад в общем списке записей истории сеансов для текущего проходимого элемента навигации.

Если предыдущей страницы нет, ничего не делает.

history.forward()✔MDN

Продвигается на один шаг вперед в общем списке записей истории сеансов для текущего проходимого элемента навигации.

Если следующей страницы нет, ничего не делает.

history.pushState(data, "")✔MDN

Добавляет новую запись в журнал сеансов с его классическим состоянием API истории, установленным для сериализации data. URL активной записи истории будет скопирован и использован в качестве URL новой записи.

(Второй параметр существует по историческим причинам и не может быть опущен; передача пустой строки является традиционной.)

history.pushState(data, "", url)

Добавляет новую запись в журнал сеансов, для состояния классического API истории которого задано значение сериализации data, а для его URL задано значение url.

Если текущий Document URL-адрес не может быть переписан на url, будет выдан "SecurityError" DOMException.

(Второй параметр существует по историческим причинам и не может быть опущен; передача пустой строки является традиционной.)

history.replaceState(data, "")✔MDN

Обновляет состояние классического API истории записи в истории активного сеанса до структурированного клона data.

(Второй параметр существует по историческим причинам и не может быть опущен; передача пустой строки является традиционной.)

history.replaceState(data, "", url)

Обновляет состояние классического API истории записи в истории активного сеанса до структурированного клона data и его URL-адреса до url.

Если текущий Document URL-адрес не может быть переписан на url, будет выдан "SecurityError" DOMException.

(Второй параметр существует по историческим причинам и не может быть опущен; передача пустой строки является традиционной.)

Примеры использования:  
<button id="go-back">Go back!</button>

document.getElementById("go-back").addEventListener("click", () => {

history.back();

});

history.go(-1);

history.go();

history.go(0);

<button id="go-forward">Go Forward!</button>

document.getElementById("go-forward").addEventListener("click", (e) => {

history.forward();

});

const state = { page\_id: 1, user\_id: 5 };

const url = "hello-world.html";

history.pushState(state, "", url);

const stateObj = { foo: "bar" };

history.pushState(stateObj, "", "bar.html");

history.replaceState(stateObj, "", "bar2.html");

Источники информации:  
<https://developer.mozilla.org/en-US/docs/Web/API/History_API>

https://html.spec.whatwg.org/#the-history-interface