

Browser

ΜΑΘΗΜΑ 3.5 HISTORY

ΠΕΡΙΕΧΟΜΕΝΑ:

1. Το αντικείμενο history
 1. Η μέθοδος pushState()
 2. Το event popstate
 3. Η μέθοδος replaceState()
 4. Τίτλος Σελίδας και Παρατηρήσεις

History API:

- Υλοποιείται από το **window.history**
- Επιτρέπει να χειριζόμαστε το ιστορικό του browser, χωρίς να φορτώνουμε εκ νέου τη σελίδα
- Αυτό είναι πολύ χρήσιμο σε SPAs (= Single Page Applications, π.χ. React, Angular, Vue), όπου οι ενημερώσεις δυναμικού περιεχομένου, μπορεί να αλλοιώσουν την εμφάνιση του URL και του ιστορικού.
 - Π.χ. αντικαθιστά τις ενημερώσεις όπου φορτώνεται εκ νέου η σελίδα με κομψό τρόπο.

Το αντικείμενο history:

- Είναι μέλος του αντικειμένου **window**
- Περιέχει τις ακόλουθες συναρτήσεις για απλή πλοήγηση με βάση το ιστορικό της σελίδας:

Property	Επεξήγηση
back()	Προηγούμενη σελίδα (αντίστοιχο με το να πατήσουμε το «Back»)
forward()	Επόμενη σελίδα (αντίστοιχο με το να πατήσουμε το «Forward»)
go(n)	Μετάβαση κατά n σελίδες (n>0: Forward, n<0: Back)

Παράδειγμα 1: history

```
<button id="backBtn">Go Back</button>
<button id="forwardBtn">Go Forward</button>
<button id="goBtn">Go to Specific Step</button>
```

```
// Example of navigating back in history
document.getElementById('backBtn').addEventListener('click', function() {
  history.back(); // Equivalent to pressing the browser's back button
});

// Example of navigating forward in history
document.getElementById('forwardBtn').addEventListener('click', function() {
  history.forward(); // Equivalent to pressing the browser's forward button
});

// Example of navigating to a specific step in history
document.getElementById('goBtn').addEventListener('click', function() {
  history.go(-2); // Go back two steps in the history
});
```



ΜΑΘΗΜΑ 3.5: history

1.1. Η μέθοδος pushState()

Η μέθοδος pushState():

- Η μέθοδος pushState() του αντικειμένου **history**:

Μέθοδος	Επεξήγηση
pushState(state, title, url)	Προσθέτει μία νέα εγγραφή στο ιστορικό του browser

- Τα ορίσματά της είναι:
 - state**: Αντικείμενο που περιέχει δεδομένα (δικά μας), χρήσιμα για να επαναφέρουμε τη σελίδα
 - (ο browser δεν κάνει refresh για μία σελίδα που δημιουργήθηκε με την pushState())
 - title**: Αναπαριστά τον τίτλο της σελίδας (αλλά συνήθως δεν χρησιμοποιείται από τους σύγχρονους browsers)
 - url**: Το url που φαίνεται στη γραμμή διευθύνσεων (σχετικό ή απόλυτο)
- Κάθε φορά που γίνεται pushState():
 - Μία καινούργια εγγραφή δημιουργείται στο history (που φαίνεται σαν να είναι ακόμα μια σείδα)**

Σημείωση:

- Για να δουλέψει ολοκληρωμένα το παράδειγμα απαιτείται να οριστεί και το event popstate (επόμενη διαφάνεια)
- Στο παράδειγμα βλέπουμε πως εναλλάσσονται οι «σελίδες» χωρίς να ενημερώνεται ακόμη το ιστορικό.

Παράδειγμα 2: pushState

```

<!-- Navigation Buttons -->
<button id="page1Btn">Go to Page 1</button>
<button id="page2Btn">Go to Page 2</button>

<!-- Content Div where the page content will be dynamically displayed -->
<div id="content">
  <p>Welcome to the app! Click a button to load content.</p>
</div>

// Function to update the content and state
function updateContent(page, content, url) {
  // Update the content div
  document.getElementById('content').innerHTML = content;

  // Push the new state to the history
  history.pushState({ page }, "", url);
}

// Event listener for Page 1 button
document.getElementById('page1Btn').addEventListener('click', function() {
  updateContent('page1', '<h2>Page 1</h2><p>This is the content of Page 1.</p>',
    'page1');
});

// Event listener for Page 2 button
document.getElementById('page2Btn').addEventListener('click', function() {
  updateContent('page2', '<h2>Page 2</h2><p>This is the content of Page 2.</p>',
    'page2');
});
    
```



ΜΑΘΗΜΑ 3.5: history

1.2. To event popstate

To event popstate:

- Ορίζεται στο window
- Ενεργοποιείται όταν ο χρήστης πατήσει το back/forward στον browser
- Το αντικείμενο event έρχεται αρχικοποιημένο με το property:

Property	Επεξήγηση
state	Το αντικείμενο - κατάσταση που είχαμε σώσει με την pushState()

- Με χρήση των πληροφοριών που είχαμε αποθηκεύσει στην κατάσταση, μπορούμε να επαναφέρουμε προγραμματιστικά τη σελίδα.

Σημείωση:

- Για να δούμε το παράδειγμα να εκτελείται, πρέπει να τρέξουμε τη σελίδα από web server.
- Ένας απλός τρόπος είναι μέσω της python με την εντολή:
 - **python -m http.server**
 - Με την οποία κατασκευάζεται ένας απλός HTTP Server και έτσι ξεπερνιέται το πρόβλημα CORS (Cross-Origin Resource Sharing) που προκαλείται επειδή το url δεν είναι έγκυρο (file://)

Παράδειγμα 3: popstate

```
<!-- Navigation Buttons -->
<button id="page1Btn">Go to Page 1</button>
<button id="page2Btn">Go to Page 2</button>

<!-- Content Div where the page content will be dynamically displayed -->
<div id="content">
  <p>Welcome to the app! Click a button to load content.</p>
</div>
```

```
// Handle back/forward button navigation
window.addEventListener('popstate', function(event) {
  if (event.state) {
    // Check which page we are navigating to
    const page = event.state.page;
    const content = page === 'page1' ?
      '<h2>Page 1</h2><p>This is the content of Page 1.</p>' :
      '<h2>Page 2</h2><p>This is the content of Page 2.</p>';

    // Update the content to reflect the history state
    document.getElementById('content').innerHTML = content;
  }
});
```

Η μέθοδος replaceState():

- Η μέθοδος replaceState() του αντικειμένου **history**:

Μέθοδος	Επεξήγηση
replaceState(state, title, url)	Αντικαθιστά την τρέχουσα εγγραφή του history (Δεν δημιουργεί νέα)

- Τα ορίσματά της είναι τα ίδια με την pushState()

Σημείωση:

- Η μέθοδος είναι χρήσιμη όταν θέλουμε να τροποποιήσουμε την κατάσταση, χωρίς να δημιουργούμε ιστορικό (όπως η pushState())
- Π.χ. μπορεί να έχουμε ένα μέρος της σελίδας με tabs, όπου θέλουμε να θυμόμαστε ποιο tab έχει πατήσει ο χρήστης, αλλά δεν θέλουμε ξεχωριστή εγγραφή στο ιστορικό για κάθε tab.

Παράδειγμα 4: replaceState

```
<!-- Navigation Tabs -->
<div id="tabs">
  <button id="overviewTab">Overview</button>
  <button id="settingsTab">Settings</button>
  <button id="activityTab">Activity</button>
</div>

<!-- Content Area -->
<div id="content">
  <h2>Welcome</h2>
  <p>Select a tab to view profile details.</p>
</div>
```



```
// Function to update the content and replace the current state
function updateProfileTab(tab, content, url) {
  // Update the content area with the selected tab's content
  document.getElementById('content').innerHTML = content;

  // Replace the current state with the new tab state and URL
  const state = { tab: tab };
  history.replaceState(state, "", url); // Replacing current history entry
}

// Event listeners for each tab
document.getElementById('overviewTab').addEventListener('click', function() {
  updateProfileTab('overview', '<h2>Overview</h2><p>This is the overview of your profile.</p>', '/profile/overview');
});

document.getElementById('settingsTab').addEventListener('click', function() {
  updateProfileTab('settings', '<h2>Settings</h2><p>Here you can change your settings.</p>', '/profile/settings');
});

document.getElementById('activityTab').addEventListener('click', function() {
  updateProfileTab('activity', '<h2>Activity</h2><p>Here is your recent activity.</p>', '/profile/activity');
});

// Handle browser back/forward button navigation with popstate
window.addEventListener('popstate', function(event) {
  ...
});

// On page load, check if there's a state and load the corresponding content
window.addEventListener('load', function() {
  ....
});
```

Τίτλος Σελίδας:

- Είδαμε ότι το όρισμα title των `replaceState()` και `pushState()` δεν δουλεύει (διατηρείται μόνο για λόγους συμβατότητας).
- Ο μόνος τρόπος να τροποποιήσουμε τον τίτλο είναι μέσω της `document.title` (οπότε και θα εμφανίζεται σωστά στο ιστορικό)

Παράδειγμα 5: replaceState

```
// Function to update the content and state
function updateContent(page, content, url) {
  // Update the content div
  document.getElementById('content').innerHTML = content;

  // Push the new state to the history
  history.pushState({ page }, "", url);

  document.title = page;
}
```

Παρατηρήσεις:

- Η υπερβολική χρήση του `pushState()`, μπορεί να επιβαρύνει το ιστορικό και οδηγούν σε δυσάρεστη εμπειρία χρήστη. Η `pushState()` πρέπει να χρησιμοποιείται για μεγάλα updates και η `replaceState()` για μικρότερα.
- Ο ορισμός τίτλου στη σελίδα είναι απαραίτητος (αλλιώς πρακτικά ο χρήστης δεν μπορεί να χρησιμοποιήσει το ιστορικό του)
- Τα δεδομένα που βάζουμε στην κατάσταση πρέπει να είναι διαχειρίσιμα, αλλιώς η διόρθωση με βάση την κατάσταση μπορεί να γίνει πολύ δύσκολη.

Όλα τα παραπάνω γίνονται με αρκετά πιο κατανοητό τρόπο στα frameworks (React, Vue, Angular)