ΠΕΡΙΕΧΟΜΕΝΑ:

1. navigator
    1. Πληροφορίες Σύνδεσης
2. permissions
    1. Γεωγραφική Θέση
    2. Πρόσβαση στην Κάμερα
    3. Πρόσβαση στο Μικρόφωνο
    4. Notifications
    5. Clipboard

ΜΑΘΗΜΑ 3.4: navigator

1. Το αντικείμενο navigator

Browser psounis

Το αντικείμενο navigator:

- Είναι μέλος του αντικειμένου **window**
- Παρέχει πληροφορίες για τον browser και την πλατφόρμα του χρήστη
- Περιέχει μέλη όπως τα ακόλουθα:

Property	Επεξήγηση
userAgent	Πληροφορίες για τον browser και την έκδοσή του
language	Η προτίμηση γλώσσας του χρήστη
platform	Πληροφορίες για το λειτουργικό σύστημα του χρήστη

Σημείωση:

- Άλλα properties που τώρα είναι πλέον παρωχημένα και διατηρούνται για λόγους συμβατότητας:

Property	Επεξήγηση
appName	Το όνομα του browser
appVersion	Πληροφορίες browser
cookieEnabled	Αν επιτρέπονται cookies

Παράδειγμα 1: navigator

```
console.log("User Agent: " + navigator.userAgent);
console.log("Language: " + navigator.language);
console.log("Platform: " + navigator.platform);
console.log("Are cookies enabled? " + navigator.cookieEnabled);
console.log("App Name: " + navigator.appName);
console.log("App Version: " + navigator.appVersion);
console.log("Are cookies enabled? " + navigator.cookieEnabled);
```

```
User Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrom
Safari/537.36
Language: en-US
Platform: Win32
Are cookies enabled? true
App Name: Netscape
App Version: 5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/129
Safari/537.36
Are cookies enabled? true
```

Πληροφορίες Σύνδεσης:

- Βλέπουμε αν ο χρήστης είναι σε σύνδεση με το property του **navigator**:

Property	Επεξήγηση
onLine	Boolean (ο χρήστης είναι σε σύνδεση)

Υπενθύμιση:

- Τα events online, offline του window, μπορούν να χρησιμοποιηθούν για να ελεγχθεί αν ο χρήστης είναι σε σύνδεση.

Το Network Information API:

- Πρόσθετα το αντικείμενο **navigator.connection** παρέχει πληροφορίες για την σύνδεση. Περιέχει τα properties:

Property	Επεξήγηση
type	Τύπος Σύνδεσης (wifi, cellular, ethernet)
effectiveType	2g, 3g, 4g κ.λπ.
downlink	Εκτιμώμενη ταχύτητα download σε Mbps
rtt	round-trip time (σε ms)

- Προσοχή ότι το παραπάνω API είναι σε πειραματική έκδοση και δεν υποστηρίζεται από όλους τους browser

Παράδειγμα 2: connection

```
// Log the online/offline status
console.log("Online status: " + (navigator.onLine ? "Online" : "Offline"));

// Add event listeners to log when the user goes online or offline
window.addEventListener('online', () => {
  console.log('You are back online.');
});

window.addEventListener('offline', () => {
  console.log('You are offline.');
});

// Check if the Network Information API is supported and log connection details
if ('connection' in navigator) {
  const connection = navigator.connection;
  console.log("Type: " + connection.type);
  console.log("Connection type: " + connection.effectiveType);
  console.log("Downlink speed: " + connection.downlink + " Mbps");
  console.log("Round-trip time (RTT): " + connection.rtt + " ms");

  // Optional: Log when the connection type changes (if supported)
  connection.addEventListener('change', () => {
    console.log("Connection type changed: " + connection.effectiveType);
  });
} else {
  console.log("Network Information API is not supported in this browser.");
}
```

Permissions API:

- Με αυτό το API που υλοποιείται στο αντικείμενο **window.navigator.permissions**, έχουμε πρόσβαση σε μέρη που απαιτούν την άδεια του χρήστη, όπως τα ακόλουθα:

Property	Επεξήγηση
Geolocation	Τοποθεσία του Χρήστη
Notifications	Notifications σε επίπεδο συστήματος
Camera	Πρόσβαση στην κάμερα του χρήστη
Microphone	Πρόσβαση στο μικρόφωνο του χρήστη
Clipboard	Αν επιτρέπεται πρόσβαση στο clipboard
Media Devices	Πρόσβαση σε συσκευές κάμερας και μικροφώνου
Bluetooth	Επικοινωνία μέσω blootooth
Sensors	Όπως accelerometers, gyroscopes κ.α.
Storage	Αποθήκευση για χρήση offline

- Βλέπουμε την άδεια για ένα χαρακτηριστικό με την ασύγχρονη μέθοδο του **permissions**:

Μέθοδος	Επεξήγηση
query({name: feature})	Επιστρέφει Promise που επιλύεται σε αντικείμενο PermissionStatus (που περιέχει τη συμβολοσειρά state με τιμές: granted, denied, ή prompt)

  - Οι τιμές είναι οι προφανείς (αν ο χρήστης έχει ερωτηθεί ήδη ή όχι και έχει απαντήσει καταφατικά ή αρνητικά)

Παράδειγμα 3: permissions-info

```
// Function to check and log permission status for a specific feature
function checkPermission(feature) {
  navigator.permissions.query({ name: feature })
    .then(permissionStatus => {
      console.log(`${feature} permission status: ${permissionStatus.state}`);
    })
    .catch(error => {
      console.log(`${feature} permission is not supported or cannot be checked: ${error.message}`);
    });
}
// Features to check with Permissions API
const features = [
  'geolocation',        // Location access
  'notifications',      // Notifications
  'camera',             // Camera access
  'microphone',         // Microphone access
  'clipboard-read',     // Clipboard read
  'clipboard-write',    // Clipboard write
  'push',               // Push notifications
  'midi',               // MIDI device access
  'background-sync',    // Background synchronization
  'persistent-storage', // Persistent storage (for offline apps)
  'ambient-light-sensor', // Ambient light sensor (sensors API)
  'accelerometer',      // Accelerometer (sensors API)
  'gyroscope',          // Gyroscope (sensors API)
  'magnetometer'        // Magnetometer (sensors API)
];
// Check permissions for each feature
features.forEach(feature => checkPermission(feature));
```

```
geolocation permission status: prompt
notifications permission status: prompt
camera permission status: prompt
microphone permission status: prompt
clipboard-read permission status: prompt
clipboard-write permission status: granted
midi permission status: prompt
background-sync permission status: granted
persistent-storage permission status: prompt
accelerometer permission status: granted
gyroscope permission status: granted
magnetometer permission status: granted
```

Geolocation API:

- Με αυτό το API που υλοποιείται στο αντικείμενο **window.navigator.geolocation**, έχουμε πρόσβαση στην τοποθεσία του χρήστη.
- Το αντικείμενο περιέχει τη μέθοδο:

Μέθοδος	Επεξήγηση
getCurrentPosition (position=>…, error=>…)	Η συνάρτηση-όρισμα αρχικοποιεί το αντικείμενο position

coords, με μεθόδους:

Μέθοδος	Επεξήγηση
latitude	Γεωγραφικό Πλάτος
longitude	Γεωγραφικό Μήκος
accuracy	Ακρίβεια

Σημείωση:

- Δεν καλούμε ρητά κάποια μέθοδο για να ζητήσουμε άδεια πρόσβασης.
- Όταν καλείται κάποια μέθοδος (όπως η getCurrentPosition()) που απαιτεί άδεια, τότε ο browser εμφανίζει το απαραίτητο popup για να αποφασίσει ο χρήστης αν εκχωρεί την άδεια.

Παράδειγμα 4: geolocation

```
function requestGeolocation() {
  // Check the permission status for geolocation
  navigator.permissions.query({ name: 'geolocation' })
    .then(permissionStatus => {
      console.log(`Geolocation permission status: ${permissionStatus.state}`);

      // If permission is granted or prompt, get and log the geolocation info
      if (permissionStatus.state === 'granted' || permissionStatus.state === 'prompt') {
        navigator.geolocation.getCurrentPosition(position => {
          console.log(`Latitude: ${position.coords.latitude}`);
          console.log(`Longitude: ${position.coords.longitude}`);
          console.log(`Accuracy: ${position.coords.accuracy} meters`);
        }, error => {
          console.error(`Geolocation error: ${error.message}`);
        });
      } else {
        console.log("Geolocation access denied.");
      }

      // Listen for permission state changes
      permissionStatus.onchange = () => {
        console.log(`Geolocation permission status changed to: ${permissionStatus.state}`);
      };
    })
    .catch(error => {
      console.error("Permission query failed: " + error.message);
    });
}

// Call the function to request geolocation and log the data
requestGeolocation();
```

MediaDevices API:

- Με αυτό το API που υλοποιείται στο αντικείμενο **window.navigator.mediaDevices**, έχουμε πρόσβαση σε συσκευές Media, όπως η κάμερα και το μικρόφωνο
- Το αντικείμενο περιέχει τη μέθοδο:

Μέθοδος	Επεξήγηση
getUserMedia ({video: true})	Promise: Αν επιλυθεί (ο χρήστης επιτρέπει την πρόσβαση στην κάμερα), επιστρέφει αντικείμενο MediaStream που μπορεί να συσχετιστεί με στοιχείο video

Σημείωση:

- Η διαχείριση του MediaStream ξεφεύγει από τους σκοπούς του μαθήματος.
- Ωστόσο μπορούμε να χρησιμοποιήσουμε το property του <video>:

Μέθοδος	Επεξήγηση
srcObject	Όταν τεθεί ίσο με αντικείμενο MediaStream, το <video> παίζει το stream

Παράδειγμα 5: camera

```html
<div class="container">
  <h1>Access Your Camera</h1>
  <video id="videoElement" autoplay playsinline></video>
  <p id="errorMessage"></p>
  <button id="stopButton">Stop Camera</button>
</div>
```

```javascript
// Function to start the camera
function startCamera() {
   navigator.mediaDevices.getUserMedia({ video: true })
      .then(stream => {
         // Display the video stream in the video element
         videoElement.srcObject = stream;

         // Save the stream to stop it later
         window.cameraStream = stream;
         errorMessage.textContent = ''; // Clear any previous error message
      })
      .catch(error => {
         // Handle errors (permission denied, no camera, etc.)
         errorMessage.textContent = `Error accessing the camera: ${error.message}`;
      });
}
```

Σημείωση:

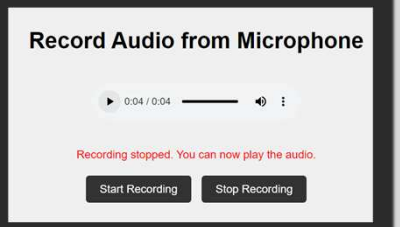
- Η πρόσβαση στο μικρόφωνο γίνεται με εντελώς αντίστοιχο τρόπο: Χρησιμοποιούμε την μέθοδο: getUserMedia({audio: true}).
- Στο παράδειγμα (που ξεφεύγει από τα όρια του μαθήματος), βλέπουμε το συνδυασμό με το MediaRecorder API για την καταγραφή της ηχογράφησης.

Παράδειγμα 6: audio

```html
<div class="container">
  <h1>Record Audio from Microphone</h1>
  <audio id="audioElement" controls></audio>
  <p id="errorMessage"></p>
  <button id="startButton">Start Recording</button>
  <button id="stopButton" disabled>Stop Recording</button>
</div>
```

```javascript
// Select elements
const audioElement = document.getElementById('audioElement');
const startButton = document.getElementById('startButton');
const stopButton = document.getElementById('stopButton');
const errorMessage = document.getElementById('errorMessage');

let mediaRecorder;
let audioChunks = [];
```

```javascript
// Function to start recording the microphone
function startRecording() {
  navigator.mediaDevices.getUserMedia({ audio: true })
    .then(stream => {
      mediaRecorder = new MediaRecorder(stream);
      // Start recording when mediaRecorder is ready
      mediaRecorder.start();
      startButton.disabled = true;
      stopButton.disabled = false;
      errorMessage.textContent = 'Recording...';

      // Collect the audio data in chunks as it becomes available
      mediaRecorder.ondataavailable = event => {
        audioChunks.push(event.data);
      };
      // When the recording stops, combine the chunks into a Blob
      mediaRecorder.onstop = () => {
        const audioBlob = new Blob(audioChunks, { type: 'audio/wav' });
        audioChunks = [];  // Reset for future recordings

        // Create a URL for the audio Blob and set it as the source for the audio element
        const audioUrl = URL.createObjectURL(audioBlob);
        audioElement.src = audioUrl;

        errorMessage.textContent = 'Recording stopped. You can now play the audio.';
      };
    })
    .catch(error => {
      errorMessage.textContent = `Error accessing the microphone: ${error.message}`;
    });
}
// Function to stop recording the microphone
function stopRecording() {
  mediaRecorder.stop();
  startButton.disabled = false;
  stopButton.disabled = true;
}
```



Notifications API:

- Με αυτό το API που υλοποιείται στο αντικείμενο **window.Notification**, έχουμε πρόσβαση στον μηχανισμό των Notifications του browser
- Το αντικείμενο περιέχει:

Μέθοδος	Επεξήγηση
permission	Τρέχουσα κατάσταση (granted, denied ή default)
requestPersmission ()	Ζητάει την άδεια του χρήστη για να δείχνει notifications. Επιστρέφει promise
new Notification (title, options)	Δημιουργεί και προβάλλει ένα νέο notification με τον τίτλο title και τις επιλογές (π.χ. body: σώμα κειμένου, icon: εικονίδιο)
window.focus()	Φέρνει το τρέχον παράθυρο στο προσκήνιο

Σημείωση:

- Το παράδειγμα δεν θα τρέξει με τον τυπικό τρόπο που ανοίγουμε τις σελίδες
- Απαιτείται να είναι σε HTTPS server, ώστε η επικοινωνία browser - server να είναι ασφαλής (βλ. βίντεο)

Παράδειγμα 7: notifications

```html
<div class="container">
  <h1>Browser Notifications</h1>
  <button id="notifyButton">Send Notification</button>
  <p id="statusMessage"></p>
</div>
```

```js
const notifyButton = document.getElementById('notifyButton');
const statusMessage = document.getElementById('statusMessage');
if ('Notification' in window) {
    if (Notification.permission === 'default') {
        Notification.requestPermission().then(permission => {
            statusMessage.textContent = (permission === 'granted')
                ? 'Notifications are enabled.'
                : 'Notifications are denied.';
        });
    } else {
        statusMessage.textContent = (Notification.permission === 'granted')
            ? 'Notifications are already enabled.'
            : 'Notifications are denied.';
    }
    notifyButton.addEventListener('click', () => {
        if (Notification.permission === 'granted') {
            new Notification('Hello!', {
                body: 'This is a browser notification.',
                icon: 'https://via.placeholder.com/150'
            }).onclick = () => {
                window.focus();
            };
        } else {
            statusMessage.textContent = 'Notifications are not enabled.';
        }
    });
} else {
    statusMessage.textContent = 'This browser does not support notifications.';
}
```

Clipboard API:

- Με αυτό το API που υλοποιείται στο αντικείμενο **window.navigator.clipboard**, έχουμε πρόσβαση στο Clipboard του συστήματος
- Το αντικείμενο περιέχει:

Μέθοδος	Επεξήγηση
writeText(text)	Γράφει το κείμενο στο clipboard
readText()	Διαβάζει το κείμενο από το clipboard
writeData(data)	Γράφει δεδομένα (bytes, π.χ. εικόνες) στο clipboard
read()	Διαβάζει δεδομένα από το clipboard

Παρατήρηση:

- Σε αντίθεση με τα events cut-copy-paste (του window) που είδαμε στο μάθημα 2.4, τα οποία δεν απαιτούν permission
  - Εδώ ενεργοποιούνται προγραμματιστικά.
- Ο browser απαιτεί permission για τις ενέργειες Clipboard που ενεργοποιούνται προγραμματιστικά, αλλά δεν απαιτεί permission για τις ενέργειες Clipboard που ενεργοποιούνται από το χρήστη.

Παράδειγμα 8: clipboard

```html
<div class="container">
  <textarea id="inputText" placeholder="Type some text here..."></textarea>
  <button id="copyButton">Copy Text</button>
  <button id="pasteButton">Paste Text</button>
  <p id="outputText"></p>
</div>
```

```js
const inputText = document.getElementById('inputText');
const outputText = document.getElementById('outputText');
const copyButton = document.getElementById('copyButton');
const pasteButton = document.getElementById('pasteButton');

// Copy text to clipboard
copyButton.addEventListener('click', () => {
    navigator.clipboard.writeText(inputText.value).then(() => {
        alert('Text copied to clipboard!');
    }).catch(err => {
        console.error('Failed to copy text: ', err);
    });
});

// Paste text from clipboard
pasteButton.addEventListener('click', () => {
    navigator.clipboard.readText().then(text => {
        outputText.textContent = `Pasted text: ${text}`;
    }).catch(err => {
        console.error('Failed to read clipboard contents: ', err);
    });
});
```