Översikt

- Läsa och skriva information lokalt.
- Lägga till stöd för Offline, genom Application Cache.

Lektion 1: Arbeta med filer

- Hantera information om Session State genom Cookies.
- Använda Sessions Storage för att upprätthålla session.
- Hantera information mellan sessioner.
- Hantera händelser relaterade till lagring.
- Arbeta med Indexed Database API.

Hantera information om Session State genom Cookies

- Cookies:
  - Är designat för att implementera session tokens.
  - Skickas till server vid varje page request.
  - Är små filer med begränsad storlek, upp till 4 KB.
  - Är öppna för missbruk.
  - Har ingen mekanism för synkronisering eller samverkan.
- Cookies är inte designat för generellt syfte.

Använda Session Storage för att upprätthålla session

- Använd objektet sessionStorage för att lagra och hämta textinformation från session:

```
sessionStorage.setItem("minNyckel", "lite text");
var textFromSession1 = sessionStorage.getItem("minNyckel");

sessionStorage["minNyckel"] = "lite text";
var textFromSession2 = sessionStorage["minNyckel"];

sessionStorage.minNyckel = "lite text";
var textFromSession3 = sessionStorage.minNyckel;
```

- Sessionsinformation är bara tillgänglig i den session som skapar det:
  - Kommer att plockas bort när användare avslutar session när webbläsare stängs ner.

Använda Session Storage för att upprätthålla session (forts.)

- Det finns ingen specificerad maxstorlek för lagring av session:
  - Eventuella begränsningar är beroende av operativsystem, webbläsare och resurser tillgängliga på enhet där informationen lagras.
  - Information kan lagras i internminnet eller på disk.

Hantera information mellan sessioner

- Använd objektet localStorage för att lagra information bestående mellan sessioner och webbsidor:

```
localStorage.setItem("minNyckel", "lite text");
var textData = localStorage.getItem("minNyckel");

localStorage["minNyckel"] ="lite text";
var textData = sessionStorage["minNyckel"];

localStorage.minNyckel = "lite text";
var textData = sessionStorage.minNyckel;
```

- Information är bestående tills de tas bort specifikt.
- Local Storage API är i grunden samma som Session Storage API, skillnad är livslängd och mål för lagring.

.eo/ee

Blob interface

- Representerar oföränderliga information, i formen raw.
- Har ett attribut som beskriver typ av media, exempelvis ”text/plain”.
- Har metoden slice(), som returnerar en del av blob, användbart om incrementel uppladdning skall ske.

.eo/ee

Hantera händelser relaterade till lagring

- Använd storage event för att meddela webbsida att förändringar har gjorts för session och/eller lokal lagring.

```
function myStorageCallback( e ) {
    alert( "Key:" + e.key + " changed to " + e.newValue );
}
…
window.addEventListener("storage", myStorageCallback, true );
```

- Egenskaper för objektet event:
  - key - namn för värde som har blivit förändrat.
  - oldValue - ursprungsvärde.
  - newValue - det nya värdet.
  - url - källa för händelser.
  - storageArea - referens till lagring har förändrats.

.eo/ee

Arbeta med Indexed Database API

- IndexedDB tillhandahåller funktioner för att lagra strukturerad information på användares dator.
- API:et är asynkront och inkluderar följande funktioner:
  - Kan lagra flera objekt.
  - `add()`, `put()`, `get()` och `delete()` kan användas för att hantera informationen.
  - Transaktioner.
  - Queries genom att använda cursors.
  - Indexering för att snabba upp queries.

.easec

Lektion 2: Lägga till stöd för Offline, genom Application Cache

- Konfiguration av Application Cache.
- Monitorering av Application Cache.
- Trigga uppdatering genom Manifest.
- Testa om nätverksanslutning finns.
- Demonstration: Lägga till stöd för Offline för din webbapplikation.

.easec

Konfiguration av Application Cache

- Manifest fil specificerar resurser som skall mellanlagras i cache och hur dessa skall uppdateras.
- Varje webbsida som skall använda mellanlagrade resurser skall referera till manifest fil:

```
CACHE MANIFEST
# version 1
CACHE:
offline.html
cleanoffline.css
styleoffline.css

NETWORK:
offline2.html

SETTINGS:
prefer-online
```

```
<!DOCTYPE HTML>
<html manifest = "/easec.manifest">
<head>
```

.easec

Trigga uppdatering genom Manifest

- Webbsida kan fortsätta att arbeta med mellanlagrade resurser även om nyare versioner finns tillgängliga.
- För att tvinga fram uppdatering:
  - Gör en markant förändring i manifestfil.
  - Eller initiera en kontroll för uppdatering, använd funktionen `update()` och därefter används funktionen `swapCache()`.

```
applicationCache.update();
...
if (applicationCache.status == 4 ) {
    applicationCache.swapCache();
}
```

.eo/ec

Testa om nätverksanslutning finns

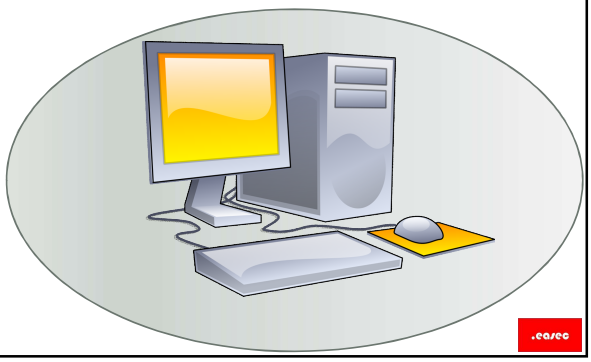
- Ibland kan det vara bättre att ta bort funktionalitet som kräver nätverksanslutning.
- Använda egenskapen `onLine` i objektet `navigator`, för att detektera status för nätverket.
- Hantera händelserna `online` och `offline` för att detektera förändringar för nätverksanslutningar.

.eo/ec

Demonstation: arbeta med Offline API

.eo/ec

Övning: Arbeta med lokal lagring och Offline API



Repetitionsfrågor