# Programowanie Internetowe


- Standard ECMA Script
- Obiektowy model dokumentu
- Data, Czas, Wyrażenia regularne
- Przechowywanie danych
- Przetwarzanie asynchroniczne

*Opracował: inż. Grzegorz Petri*

# Przegląd zagadnień

- Obiekt Date
- Format daty i czasu
- Konwersja daty
  - String to Date
  - Timestamp
- Obiekt Cookie
  - Zasady
  - Zapis
  - Odczyt

# Obiekt Date

 & 

*Data i Czas*

# Obsługa daty i czasu

*Object Date*

- Od lat format reprezentacji daty i czasu nastręcza programistom oraz twórcom treści i witryn problemów dotyczących kompatybilności i konwersji
- JavaScript udostępnia wbudowany obiekt `Date`, który przechowuje aktualną datę i godzinę oraz dostarcza dedykowanych metod dostępu do tych informacji
- Ustawiając datę obiekt jest w stanie wykryć błędną datę i ustawić poprawną*
- Zadania wspierane przez obiekt Date:
  - określanie w kalendarzu daty przyjazdu i/lub wyjazdu
  - określenie czasu rozpoczęcia i/lub zakończenia
  - określenie daty i czasu utworzenia lub wystąpienia
  - określanie pozostałego czasu

* *próba ustawienia 32 Stycznia ustawi ostatecznie 1 Lutego*

# Odczytywanie daty

*Składowe czasu*

- Obiekt można używać na dwa sposoby stosując słowo kluczowe `new Date();`
  - instancja bez argumentów zwraca obecną datę oraz czas, wg ustawień lokalnego komputera,
  - wywołując konstruktor z parametrami wskazujemy określoną datę i/lub czas.

```
let teraz = new Date();    // konstruktor bez parametrów
console.log(teraz); // wynik:
// Mon May 18 2020 18:43:52 GMT+0200 (czas środkowoeuropejski letni)
```

| *Dzień tygodnia* | *Miesiąc* | *Dzień* | *Rok* | *Godzina* | *Minuta* | *Sekunda* | *Strefa czasowa* |
|---|---|---|---|---|---|---|---|
| Mon (*Poniedziałek*) | May (*Maj*) | 18 | 2020 | 18 | 43 | 52 | GMT+0200 |

# Formaty daty

*Properties & Methods*

- Obiekt `Date` dostarcza metody dostępowe (getters) oraz ustawiające (setters)
- Formaty z jakimi współpracuje obiekt Date:
  - **`unix timestamp`** – liczba całkowita milisekund, które upłynęły od 1970-01-01
  - **`string`** – format zrozumiały dla człowieka, np.:
    - January 31 1980 12:30
    - 2018-06-16
  - **`argumenty obiektu Date`** – pozwala utworzyć obiekt daty podając poszczególne wartości we właściwej kolejności !!!

# Tabela metod obiektu Date

| *Opis* | *Metoda pobierania*** | *Metoda ustawiania* | *Zakres* | *Przykład* |
|---|---|---|---|---|
| Rok | `getFullYear()` | `setFullYear()` | `YYYY` | 1970 |
| Miesiąc | `getMonth()` | `setMonth()` | `0-11` | 0 = January (*styczeń*) |
| Dzień (*miesiąca*) | `getDate()` | `setDate()` | `1-31` | 1 = 1-szy dzień |
| Dzień (*tygodnia*) | `getDay()` | `setDay()` | `0-6` | 0 = Sunday (*niedziela*) |
| Godzina | `getHours()` | `setHours()` | `0-23` | 0 = północ |
| Minuta | `getMinutes()` | `setMinutes()` | `0-59` | |
| Sekunda | `getSeconds()` | `setSeconds()` | `0-59` | |
| Milisekunda | `getMiliseconds()` | `setMiliseconds()` | `0-999` | 100ms = 1s |
| timestamp* | `getTime()` | `setTime()` | Milisekundy* | *getTime() nie posiada getUTCTime()* |

```
let theForce = new Date();
theForce.setMonth(4); theForce.setDate(4); // jaka to będzie data?
```

** znacznik czasu – Uniksowy format czasu liczony w milisekundach od tzw. Epoki*

*** metody* `getFoo()` *obliczają czas na podstawie ustawień maszyny użytkownika, ale istnieją jeszcze metody* `getUTCFoo()` *które bazują swoje obliczania na podstawie UTC (Coordinated Universal Time)*

# Porównywanie dat

*Efektywne sposoby porównania timestampów*

Sprawdzenie obecnej daty i godziny poprzez inicjację nowego obiektu `Date`:

```
var data = new Date();
var teraz = data.getTime();
```

Zamiast inicjować nowy obiekt `Date` wystarczy wywołać szybszy odpowiednik:

```
var teraz = Date.now(); // return ms: 1589829664218
```

Następnie podać datę do porównania, np. parsując wskazaną datę do obiektu `Date`:

```
var wtedy = Date.parse("2020-02-20");// ret ms: 1582156800000
```

Porównanie dwóch dat:

```
if(teraz > wtedy){ // czy Teraz jest większe od Wtedy ?
   console.log("TAK"); // 1589829664218 > 1582156800000
} else {
   console.log("Nie"); // 1582156800000 > 1589829664218
```

# Obiekt Cookie

*Przechowywanie wartości*

# Obiekt Cookie

*Podstawy*

- Ciasteczko jest mechanizmem do przechowywania prostych wartości tekstowych w przeglądarce użytkownika odwiedzającego witrynę
- Ciasteczko posiada prostą strukturę `klucz=wartość` oraz termin przydatności
- Poza tymi podstawowymi właściwościami jest jeszcze szereg innych wpływających na bezpieczeństwo przechowywanych informacji
- Pomimo wszystko ciasteczko nie powinno przechowywać wrażliwych danych
- Ciasteczka można tworzyć oraz usuwać (*ręcznie lub automatycznie*)

```
document.cookie = "komunikat=odczytano";   // zapis 1-go ciasteczka
document.cookie = "menu=zablokuj";         // zapis 2-go ciasteczka
console.log(document.cookie); // odczyt z ciasteczek - wynik:
                    komunikat=odczytano;menu=zablokuj
```

# Obiekt Cookie

Ciasteczko oprócz podstawowej struktury `klucz=wartość` posiada szereg właściwości zapisywanych w identyczny sposób, oddzielonych średnikiem `;`

- Data wygaśnięcia ciasteczka, po której ciasteczko nie może funkcjonować (*format daty musi być zgodny z formatem standardu UTC*)
- Domena i Ścieżka w obrębie, których ciasteczko może funkcjonować (*format ścieżek i domen musi być zgodny ze standardem URL oraz nazw domenowych*)

Przykład ciasteczek z widoku Narzędzi deweloperskich przeglądarki:

| Nazwa | Wartość | Domain | Path | Wygasa / Max-Age | Rozmiar | HttpOnly | Secure | SameSite | Ostatni dostęp |
|---|---|---|---|---|---|---|---|---|---|
| do | 2020-05-25 | lo.local | /gtkWeb/gtkMake | Sesja | 12 | false | false | None | Mon, 18 May 2020 22:0... |
| od | 2020-03-25 | lo.local | /gtkWeb/gtkMake | Sesja | 12 | false | false | None | Mon, 18 May 2020 22:0... |

# Tabela właściwości obiektu Cookie

| *Własciwość* | *Znaczenie* | *Typ danych* | *Przykład* |
|---|---|---|---|
| name | Nazwa (*identyfikator dostępu*) | string* | klucz |
| value | Wartość (*treść*) | string* | wartość123 |
| domain | Domena (*limit ok 20 ciasteczek*) | string | localhost<br>www.domena.pl |
| path | Ścieżka (*w obrębie serwera/katalogu*) | string | /gtkWeb/gtkMake |
| expires | Data wygaśnięcia | string | Mon, 18 May 2020 18:43:52 UTC |
| size | Rozmiar (*wartości*) | DO-ODCZYTU | 12 |
| httpOnly | Blokuje dostęp dla JS** | boolean | true LUB false |
| secure | Zabezpieczenie (*HTTP lub HTTPS*) | boolean | true LUB false |
| sameSite | Tylko ta strona ma prawo wysyłać | string | Lax LUB Strict LUB None |
| lastModified | Ostatni dostęp (*odświeżenie*) | string | Mon, 18 May 2020 18:43:52 UTC |

**** wartości powinny być zakodowane poprzez*** `encodeURIComponent()`

***** Serwer-web ustawia ciasteczko za pomocą nagłówka HTTP Set-Cookie blokując dostęp z poziomu środowiska JavaScript przeglądarki, czyli*** `document.cookie`

# Metody zarządzania ciasteczkami

*To nie są wbudowane metody lecz własne ! ! !*

Ustawianie ciasteczka z parametrami (*fragment metody!*):

```
function setCookie( identyfikator, wartość, opcje ){
  if(navigator.cookieEnabled)
  { //czy ciasteczka są włączone
    const cookieName = encodeURIComponent(identyfikator);
    const cookieVal = encodeURIComponent(wartość);
    let cookieText = cookieName + "=" + cookieVal;
    if(typeof opcje.EXP === "number") {
       const data = new Date();
 data.setTime(data.getTime()+(opcje.EXP * 24*60*60*1000));
       cookieText += "; expires=" + data.toGMTString();
    }
    if(opcje.PRAMETR)
       cookieText += "; PARAMETR="+wartośćParametru;
    document.cookie = cookieText;
  }
}
```

# Metody zarządzania ciasteczkami

*To nie są wbudowane metody lecz własne ! ! !*

Pobieranie ciasteczka po nazwie (*fragment metody!*):

```
function getCookie( identyfikator ){
  const name = encodeURIComponent(document.cookie);
  var ckVars = name.split('%3B');  // znak ;
  for( let i=0; i<ckVars.length; i++){
     var ckPairs = ckVars[i].split('%3D'); // znak =
     var ckKey = ckPairs[0].replace('%20','');
     var ckValue = ckPairs[1];
  }
}
```

Usuwanie ciasteczka po nazwie poprzez ustawienie daty przydatności:

```
function deleteCookie( identyfikator ){
  setCookie( identyfikator, „”, {‘max-age’:-1} );
}
```

```
dialog(
  ‘Pytania?’
);
```