Kilka słów na start

1. **Dużo nauki**

   Tak, jest dużo materiału. Trzeba spędzić sporo czasu, żeby go opanować

2. **Chęci**

   Programowania można się nauczyć, ale są do tego potrzebne chęci – nic się samo nie zrobi.

3. **Ćwiczenia**

   Jest dużo ćwiczeń – pamiętajcie – ćwiczenia praktyczne są najważniejsze.

Środowisko frontowca – najważniejsze

Środowisko frontowca - przeglądarki

Środowisko frontowca - IDE

Visual Studio Code

https://code.visualstudio.com/docs/getstarted/tips-and-tricks

Home -> interactive playground

Środowisko frontowca - Inne

Debugowanie kodu

Proces debugowania polega na znalezieniu miejsca występowania błędu w naszym kodzie. Następnie dzięki temu możemy znaleźć przyczynę wystąpienia błędu oraz go poprawić. Do debugowania będziemy używać narzędzia deweloperskiego dostępnego w każdej przeglądarce. W naszym przypadku użyjemy Google Chrome.

Najprostszym a zarazem najmniej precyzyjnym sposobem debugowania kodu jest jego wykonanie i sprawdzenie w konsoli, w jakim pliku i linii został wywołany błąd, następnie następuje lokalizacja błędu i jego naprawienie.

Drugi sposób to dodanie w naszym kodzie **console.log()** w odpowiednich miejscach, aby móc w konsoli śledzić, jak przebiega wykonanie naszego skryptu.

```
function getName(name) {
    console.log('Start function getName');
    console.log('Get attr ' + name);
    name = 'Hello ' + name;
    console.log('Added greetings to name');
    return name;
}
```

Debugowanie kodu

Jeśli nasz kod jest dłuższy niż kilka linii, to musimy skorzystać z bardziej zaawansowanych narzędzi. Dzięki narzędziom debugowania możemy prześledzić aktualny stan podczas wykonywania skryptu w dosłownie każdym jego momencie.

```
var globalName = 'Tomek';

function sayMyName(name) {
    var greeting = 'Hello';
    name = greeting + ' ' + name;
    debugger;
    return name;
}
function sayGlobalName() {
    var greeting = 'Hi';
    var name = greeting + ' ' + globalName;
    return name;
}
```

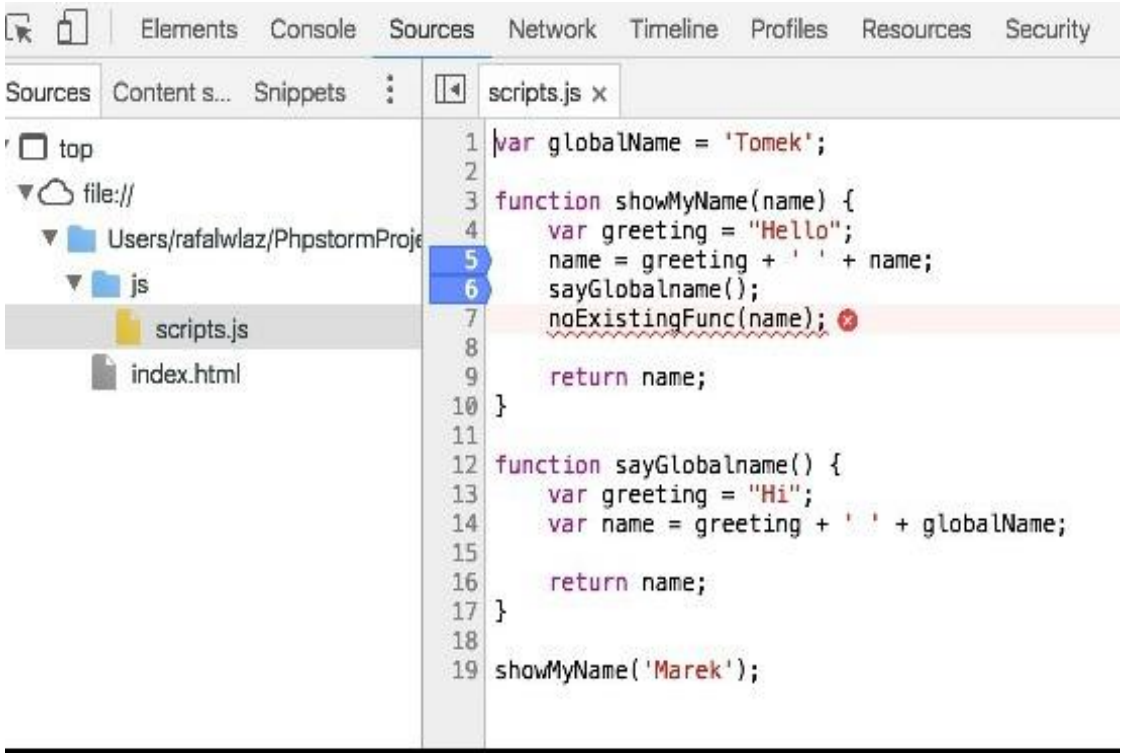
Debugowanie kodu

Jak rozpocząć proces debugowania? Należy przejść do zakładki Sources, a następnie wybrać plik JavaScript, który chcemy debugować.

Klikając na numer linii kodu, możemy oznaczyć tzw. breakpointy, czyli miejsca, gdzie wykonywanie naszego kodu się zatrzyma.

Możemy dodać ich dowolną liczbę.

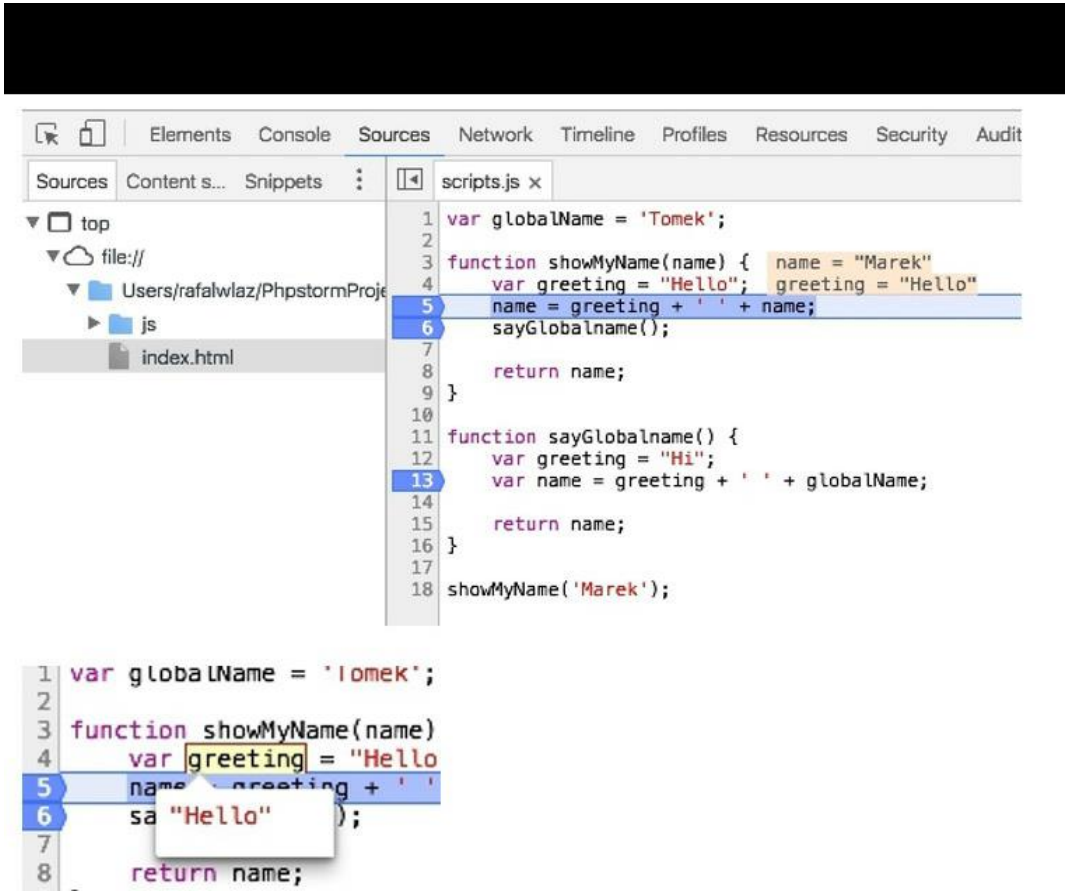
Breakpointy są zaznaczone na niebiesko.



Debugowanie kodu

Po odświeżeniu strony, skrypt rozpocznie pracę w trybie debugowania i zatrzyma się w miejscu pierwszego breakpointu a aktualna linia zostanie podświetlona.

Możemy także najechać myszą na dowolną zmienną i w dymku pojawi się jej aktualna wartość.



Debugowanie kodu

Po prawej stronie okna narzędzi deweloperskich znajduje się menu sterowania przebiegu skryptu.

Dodatkowo znajdują się tam informacje o aktualnych wartościach zmiennych w naszym skrypcie, zarówno w zakresie lokalnym (np. funkcji), jak i globalnym.

Debugowanie kodu

Pamiętajcie, że Debugger pozwala nam prześledzić sposób, w jaki kod jest wykonywany. Możemy też prześledzić, jak zmieniają się zmienne oraz sprawdzić, w jakiej kolejności kod jest wykonywany. Jest to tzw. Call Stack.

Breakpointy pozwalają zatrzymać wykonywanie skryptu w wybranym momencie, aby prześledzić aktualny stan.

Szczegółowe informacje oraz instrukcje odnośnie debuggowania w Google Chrome znajdziecie na stronie:

```
https://developers.google.com/web/tools/chromedevtools/debug/?hl=en
```

Warto odwiedzić

http://www.w3schools.com/

https://www.ecmainternational.org/memento/TC39.htm

http://www.ecma-international.org/ecma-262/6.0/

https://nodejs.org/en/

http://sekurak.pl/

http://wazniak.mimuw.edu.pl/

https://blog.codinghorror.com/

https://www.smashingmagazine.com/