Informacje pobrane ze strony:

https://www.rekinysukcesu.pl/blog/internet/chrome-devtools-najlepszy-przyjaciel-programisty-aplikacji-webowych

Konsola Chrome DevTools - najlepszy przyjaciel programisty aplikacji webowych

Konsola Chrome Devtools to zestaw narzędzi dla programistów aplikacji webowych, który jest wbudowany bezpośrednio w przeglądarkę Google Chrome. Narzędzie to służy zarówno do edycji stron internetowych, jak i szybkiej diagnozie problemów. Bez wątpienia to narzędzie jest **przyjacielem każdego programisty aplikacji webowych**.

Jak uruchomić DevTools?

Istnieje wiele sposobów otwierania Devtools:

- jeżeli chcemy pracować z DOM lub CSS należy kliknąć PPM (prawy przycisk myszy) na dowolny element mieszczący się na stronie i wybrać opcję **ZBADAJ**.

Można również przejść za pomocą skrótu klawiszy:

- dla MAC comand + option + C
- dla Windows, Lunux, Chrome OS: Control + Shift + I
- lub Control + Shift + C, wtedy możemy wybrać bezpośrednio element, który chcemy zbadać.

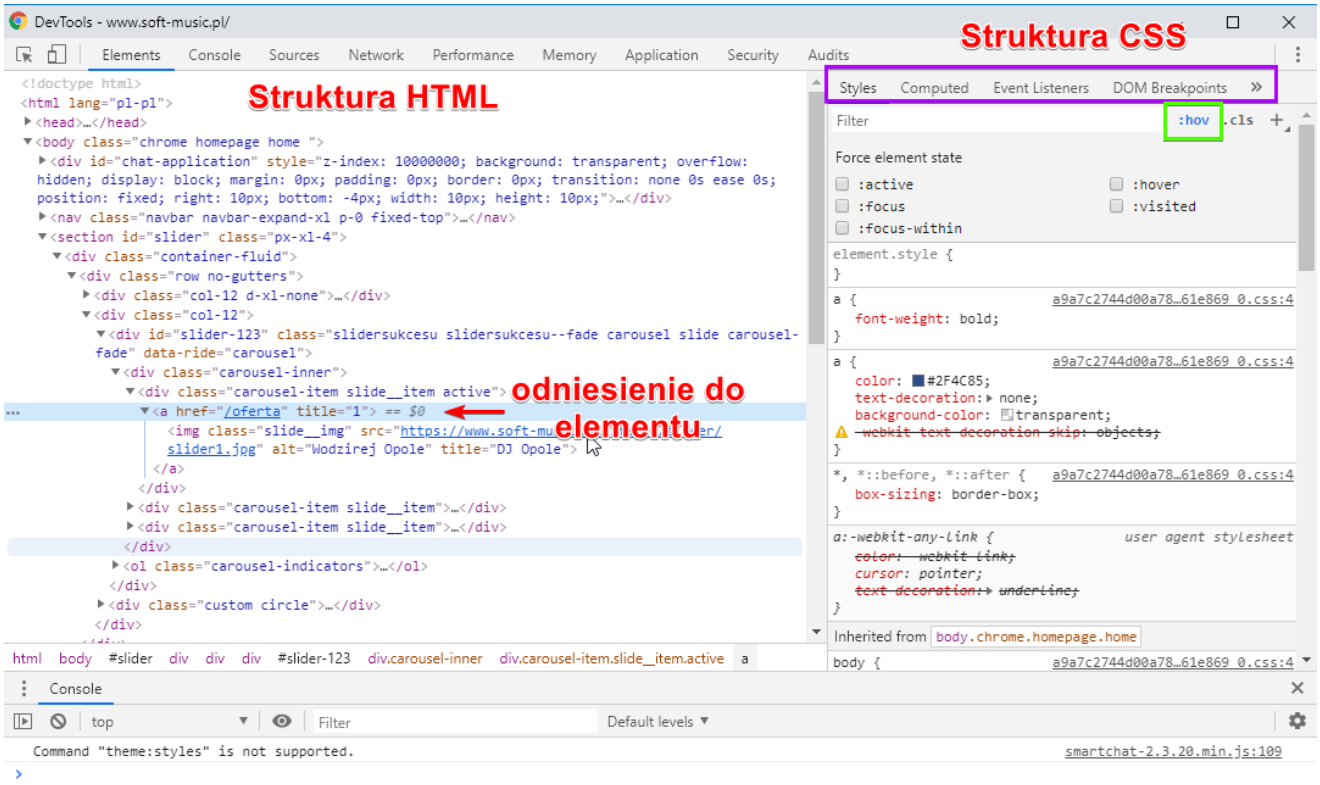
DevTools składa się z wielu komponentów

W tym artykule skupię się na opisaniu najważniejszych elementów DevTools:

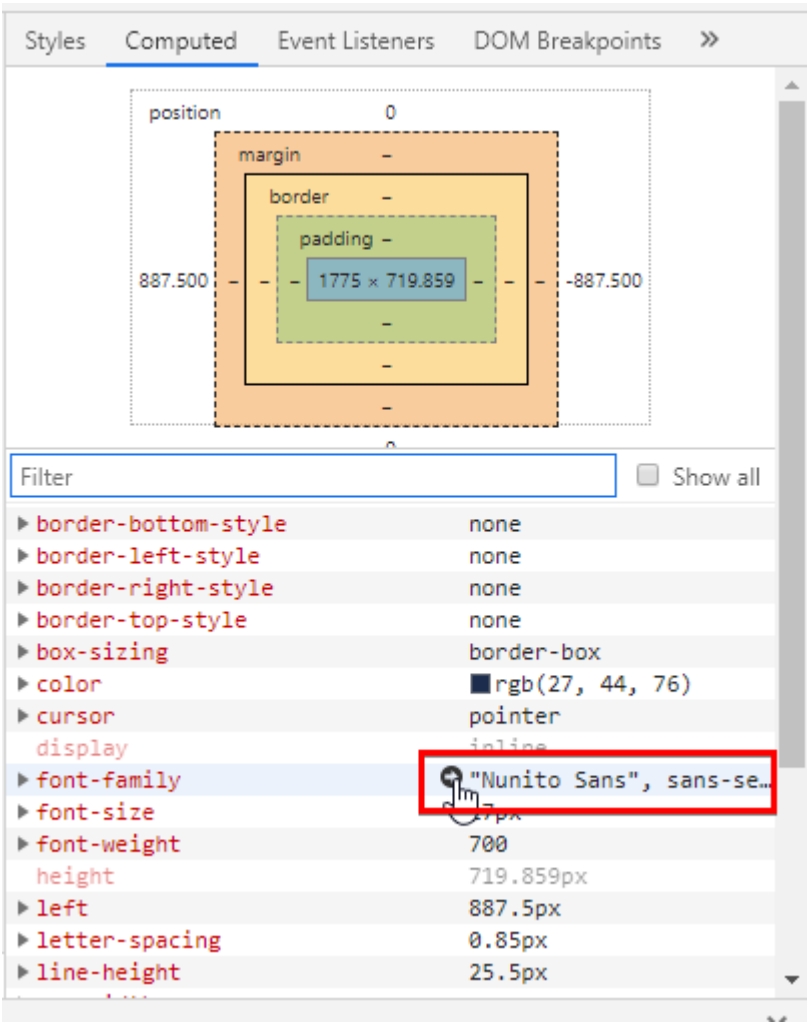
- Elements
- Console
- Network
- Performance, Memory
- Sources

Elements

Zakładka **Elements** służy do łatwej edycji poglądowej strony. Mamy tutaj strukturę HTML oraz CSS aplikacji. Do elementów można dostać się poprzez kliknięcie w strukturze HTML. W tym momencie po prawej stronie pokazują nam się style CSS przypisane do tego elementu.



Computed — pokazuje aktualne ustawienia elementów oraz pełną listę właściwości przypisanych do tego elementu HTML, klikając w strzałeczkę przy parametrze, przechodzimy do ustawień, gdzie jest ustawiony atrybut CSS.

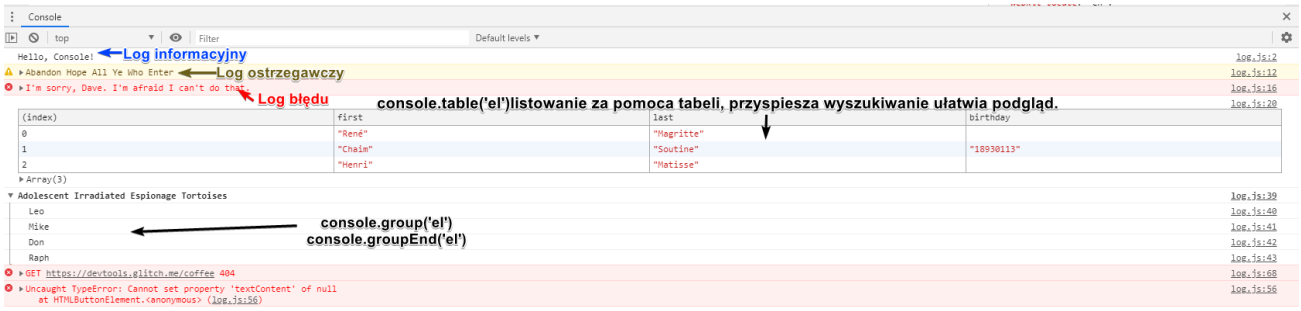


:hov - służy do wymuszania stanów danego elementu, dzięki temu mamy możliwość podglądu stylu dla szukanego stanu. Mamy do wybrania następujące stany:

- active — spełnia zasadę pseudoklasy :active od momentu przytrzymania LPM (lewego przycisku myszy) nad wybranym obszarem do momentu puszczenia.
- hover — spełnia zasadę pseudoklasy :hover, gdy w danym momencie nad obszarem elementu znajduje się kursor myszy
- focus — spełnia zasadę pseudoklasy :focus, gdy w danym momencie użytkownik skupił się na danym elemencie (np. za pomocą klawisza TAB)
- visited — spełnia zasadę pseudoklasy :visited, gdy w danym momencie element jest odwiedzonym linkiem przez użytkownika.

Console

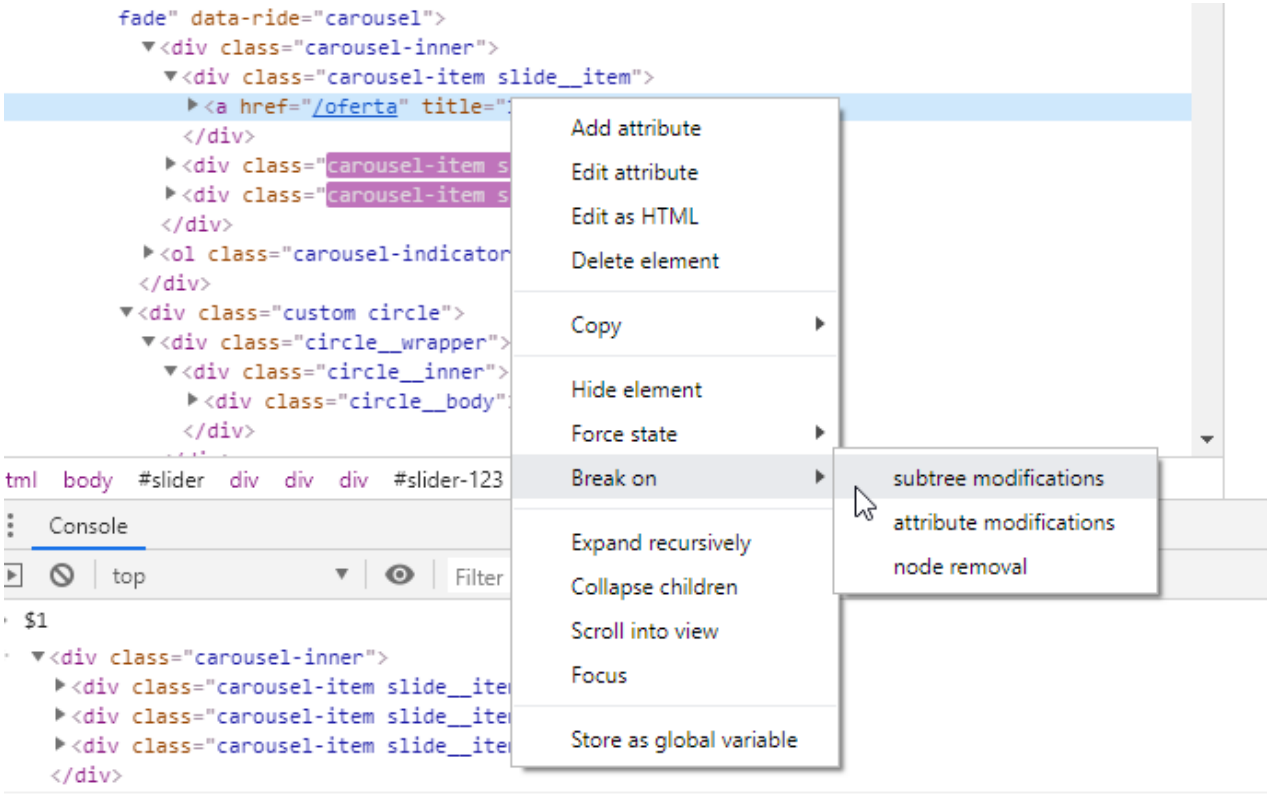
Służy do pisania kodu lub podglądu zarejestrowanych komunikatów, aby upewnić się, że kod działa zgodnie z oczekiwaniami.



Konsola posiada różnego typu opcje wyświetlania komunikatów:

- console.info — różni sie od console log tym, że jest ikona przy listowaniu;
- console.group — pozwala na grupowaniu komunikatów, rozpoczyna się console.group('info'), a kończy console.groupEnd('info');
- console.groupCollapsed — pozwala zgrupowane komunikaty, jednocześnie jest przedstawiony za pomocą zwiniętych informacji;
- console.table — widok listowania logu jest przedstawiony za pomocą tabeli;
- console.time('start') - pozwala na mierzeniu czasu dla wykonywanego kodu, rozpoczyna się console.time('test'), a kończy console.timeEnd('test').

W konsoli możemy odwoływać się do elementów za pomocą $0 (element zaznaczony), $1, $2.. (elementy podrzędne) — są to obiekty, możemy więc dostać się bezpośrednio do ich pól, metod.



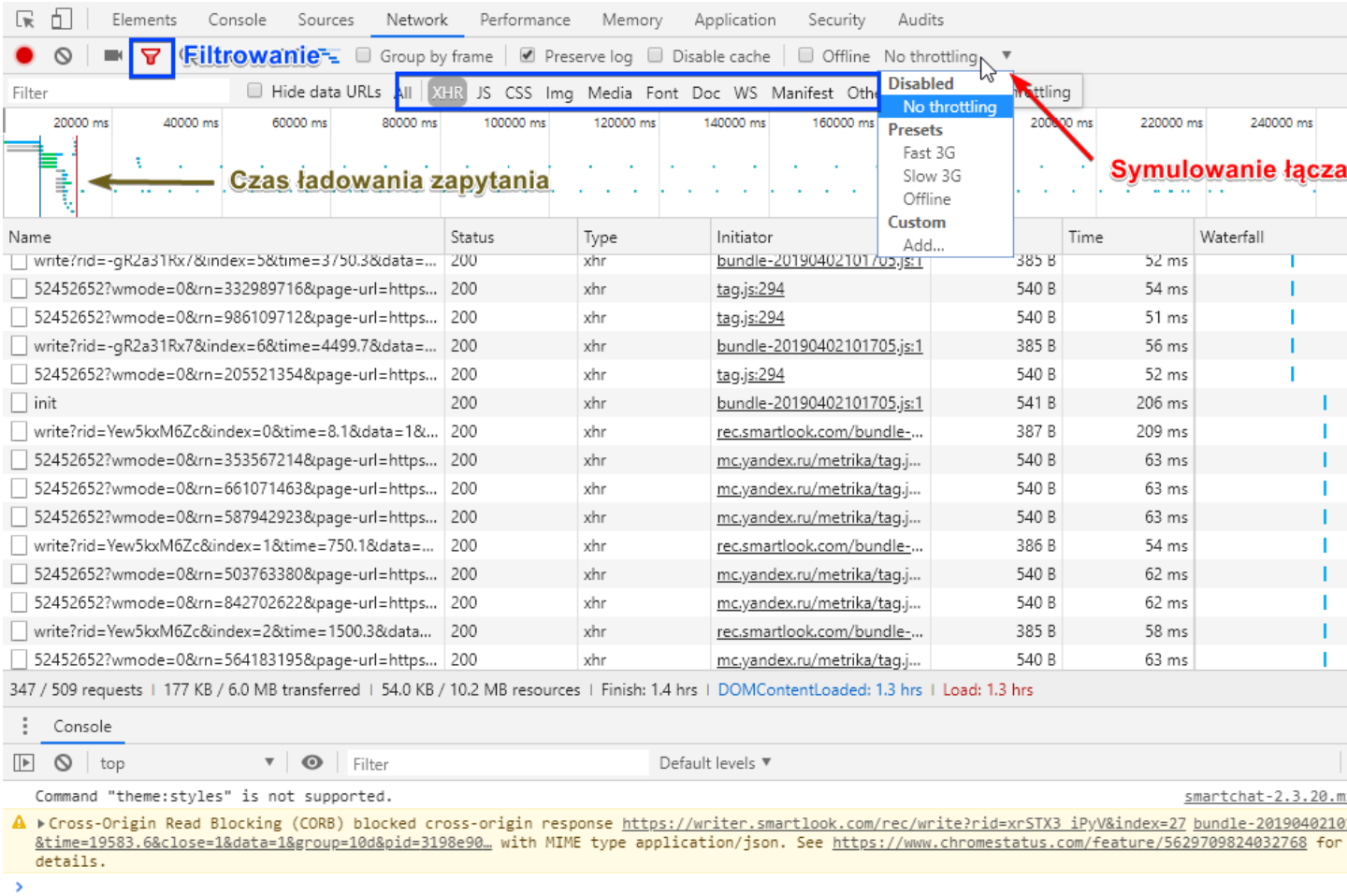
break-on -> subtree modifications — przerwanie przy debugowaniu, w momencie, gdy wybrany element zostanie zmodyfikowany, kod zostanie przerwany, dzięki temu łatwo znaleźć miejsce wystąpienia błędu.

Network

Network wykorzystujemy głównie do weryfikacji, czy zasoby pobierane oraz przesyłane są zgodne z oczekiwaniami. Najczęściej korzystamy z niej, gdy:

- upewniamy się, czy requesty są pobierane i wysyłane
- sprawdzamy właściwości pojedynczego zapytania — nagłówki, zawartość, wynik zapytania.
- sprawdzamy czas wczytywania zasobów (możemy zobaczyć czas ładowania rozbity na poszczególne typy dziedziny czasu — zapytanie, TTFB — wartością podawaną w mikrosekundach mierzoną od momentu wysłania zapytania, do chwili otrzymania przez użytkownika pierwszego bajtu danych wysłanych przez serwer)

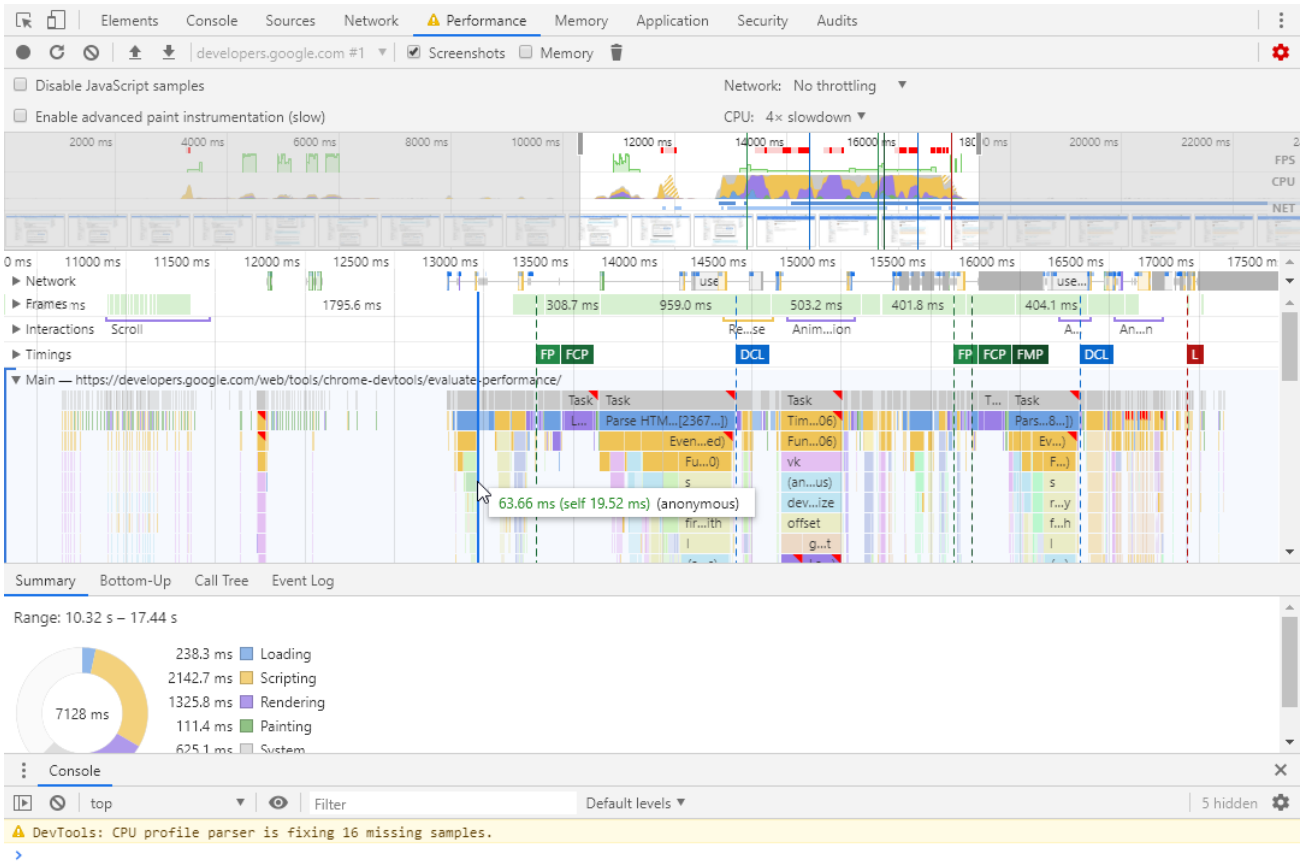
Networki w łatwy sposób możemy filtrować, należy wybrać filtruj — zaznaczony lejek poniżej. Umożliwia nam to filtrowanie rodzaju zapytań (XHR, JS, CSS, Img itd...). DevTools posiada również opcji symulowania prędkości łącza — zaznaczono na poniżej.



Performance, Memory

Przy tworzeniu aplikacji istotne jest tworzenie optymalnego kodu, żeby aplikacja nie była przeciążona. Do tego wykorzystuj symulację oraz badaj pamięć, aby zapobiec wyciekom pamięci i obniżeniu wydajności aplikacji.

Szczegółowe zasady analizy pod kątem zużycia CPU oraz pamięci opisuje Google.



Sources

Znajdziemy tutaj pełnoprawny edytor, w którym możemy wykonywać edycję czy debugowanie. Możemy tutaj dodać projekt lokalny poprzez wybranie Filesystem -> Add folder to workspace, teraz należy wybrać folder projektu oraz zezwolić na dostęp. Wszystkie pliki w tym momencie są dostępne z poziomu DevTools. Przyśpiesza nam to pracę przy debugowaniu. W przypadku, gdy mamy tablicę obiektów składającą się z 1000 elementów niewygodne jest przechodzenie krok po kroku, aż do spełnienia warunku. W tym momencie możemy napisać:

```
if(user.name==”rekin”) debugger;
```

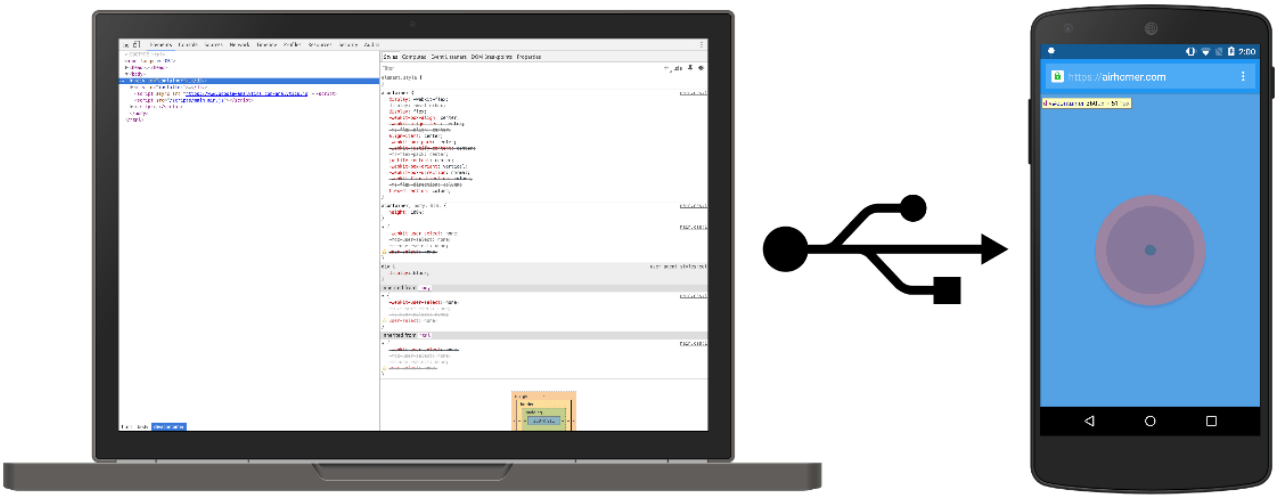
Debugger zostanie wywołany w momencie, gdy warunek zostanie spełniony.

Testowanie responsywności

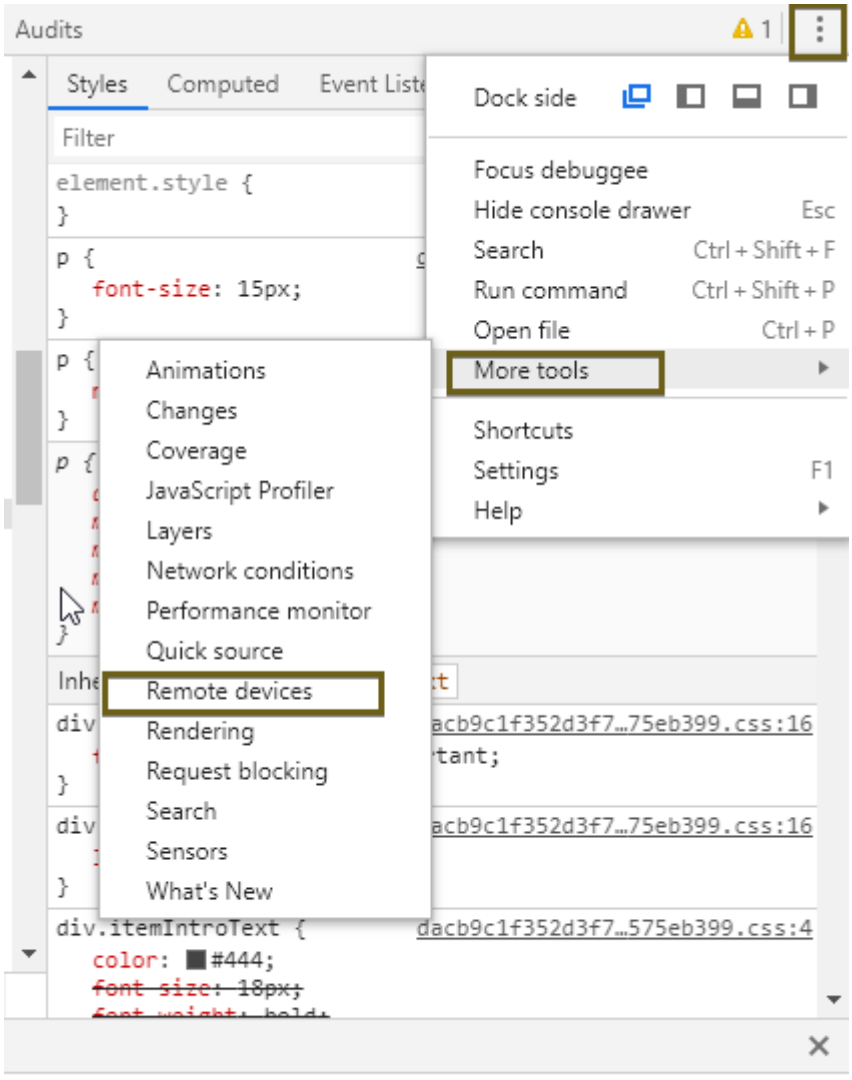
Responsywność aplikacji możemy badać w 2 sposoby:

- poprzez przeglądarkę -> sprawdź artykuł Mateusza poświęcony RWD
- poprzez telefon.

Do badania poprzez telefon potrzebujemy telefonu podłączonego do komputera oraz odpowiedniej przeglądarki. Dla telefonu z systemem Android będzie to oczywiście Google Chrome, dla iPhone'ów - Safari.



źródło: https://developers.google.com/web/tools/chrome-devtools/remote-debugging/
Należy wybrać Customize and control DevTools -> More tools -> Remote devices.



Jeżeli mamy podłączony telefon, należy ustawić port i adres serwera, następnie otworzyć stronę. Szczegółową instrukcję znajdziecie w dokumentacji Google. Warto tutaj zaznaczyć, że **taki sposób badania aplikacji pozwala nam uniknąć błędów symulatora Google Chrome**, ponieważ korzystamy z fizycznego urządzenia.

Najważniejsze skróty w DevTools - konsoli Google Chrome

- Control + Shift + I — otworzenie okna DevTools,
- Control + Shift + C — otworzenie okna DevTools z wybranym bezpośrednio, elementem, który chcemy zbadać,
- Control + Shift + M — przełączenie widoku do badania responsywności,
- Control + Shift + J — włączenie widoku konsoli,
- Control + K — czyszczenie konsoli,
- Control + F — otworzenie wyszukiwania,
- Control + [lub Control +] — przechodzenie po oknach Dev Tools (elements,console, network itd…)
- Control + Shift + P — skrót ten umożliwia szybki dostęp do opcji DevTools znany w takich edytorach jak Atom czy VS Code.
- Control + Shift + R — Twardy reset