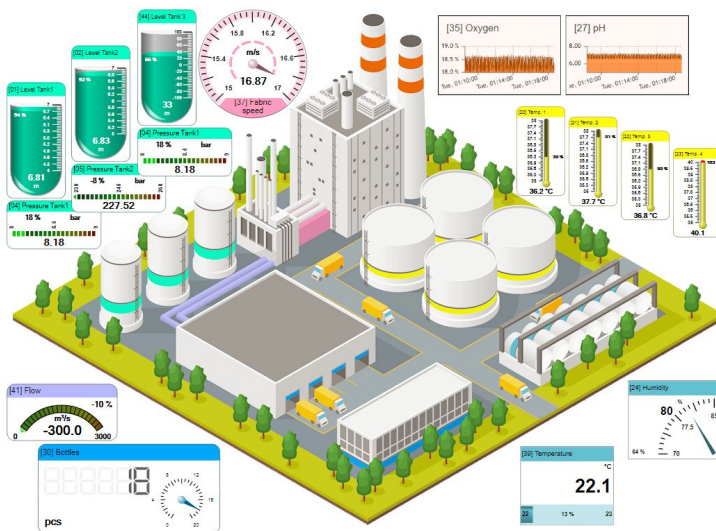


# Kurs tworzenia wizualizacji z wykorzystaniem MultiCon Sidgets

- Wersja firmware MultiCon: od **5.14.0**
- Do współpracy z urządzeniami serii MultiCon

## MultiCon Sidgets



Przed rozpoczęciem użytkowania oprogramowania należy dokładnie zapoznać się z niniejszym kursem. Producent zastrzega sobie prawo wprowadzania zmian bez uprzedzenia.

## **SPIS TREŚCI**

1. WPROWADZENIE.....	3
2. TEST DZIAŁANIA SIDGETÓW.....	4
2.1. Wstęp.....	4
2.2. Przygotowanie infrastruktury.....	4
2.3. Przygotowanie strony testowej.....	4
3. PRZYGOTOWANIE ŚRODOWISKA DO TWORZENIA WIZUALIZACJI.....	8
3.1. Wstęp.....	8
3.2. Instalacja Visual Studio Code.....	8
3.3. Uruchomienie Visual Studio Code.....	9
3.4. Dostosowanie Visual Studio Code.....	10
4. TWORZENIE WIZUALIZACJI.....	14
4.1. Wstęp.....	14
4.2. Podstawowa struktura wizualizacji.....	14
4.3. Atrybuty silnika sidgetów.....	15
4.4. Parametry sidgetów.....	17
4.5. Sposoby deklaracji parametrów sidgetów.....	23
4.6. Tworzenie wizualizacji w praktyce.....	24
4.7. Sidgety tekstowe.....	30
4.7.1. Wstęp.....	30
4.7.2. Deklaracja.....	30
4.7.3. Zmienne.....	32
5. KOMPATYBILNOŚĆ.....	35

Znaczenie symboli używanych w niniejszym kursie:



*Symbol ten zwraca uwagę na szczególnie istotne opisy. Zalecane jest dokładne zapoznanie się z uwagami oznaczonymi tym symbolem.*



*Symbol ten oznacza dodatkowe informacje i wyjaśnienia, które mogą się przydać, by w pełni zrozumieć omawiane zagadnienie.*

## 1. WPROWADZENIE

Technologia **MultiCon Sidgets** daje możliwość łatwego tworzenia wizualizacji typu SCADA służących do podglądu w czasie rzeczywistym pomiarów udostępnianych przez urządzenia **MultiCon**. Każda wizualizacja jest tworzona przez użytkownika w postaci strony sieci Web, przy czym w celu utworzenia prostej wizualizacji nie wymaga się od użytkownika specjalistycznej wiedzy programistycznej. Do tworzenia prostych wizualizacji wymagane jest jedynie zapoznanie się z niniejszym przewodnikiem. Technologia **MultiCon Sidgets** opracowana została dodatkowo w taki sposób, że bardziej doświadczeni w programowaniu użytkownicy, mogą ją wykorzystać jako część większego systemu informatycznego, np. z wizualizacjami dostępnymi w obszarze kont jego użytkowników.

Głównym założeniem przy tworzeniu tej technologii było zapewnienie minimalnej liczby czynności potrzebnej do wykonania przez użytkownika w celu stworzenia prostej, zmieniającej się w sposób dynamiczny, wizualizacji procesu przemysłowego, do której uruchomienia można by było wykorzystać dowolne urządzenie posiadające możliwość przeglądania sieci Internet za pomocą przeglądarki internetowej.

Jako, że technologia ta działa zgodnie z najnowszymi standardami budowy stron WWW takich jak HTML5+, CSS3+, JS6+, to do jej obsługi wymagane jest użycie nowszych wersji przeglądarek internetowych<sup>1</sup>, przy czym nie ma tutaj znaczenia platforma sprzętowa, czy wykorzystywany system operacyjny, na którym użytkownik będzie chciał tę wizualizację wyświetlić. Umożliwienie dostępu do urządzeń **MultiCon** poprzez sieć Internet i umieszczenie przygotowanej wizualizacji na dostępnym publicznie serwerze pozwoli dodatkowo na nadzór monitorowanych obiektów z dowolnego miejsca na świecie.



**sidget** (*wym. sidżet*) – umieszczany na stronach www obiekt graficzny, służący do wizualizacji w czasie rzeczywistym pomiarów zmieniających się w urządzeniach z serii **MultiCon**. Graficzna reprezentacja tego obiektu może przypominać wyglądem fizyczne urządzenie pomiarowe, takie jak miernik wskazówkowy, dioda, linijka diodowa, rejestrator wykresu, itp. Sidget nie wymaga ustawiania podstawowych parametrów wizualizowanego kanału, gdyż są one na bieżąco pobierane wprost z urządzenia **MultiCon**.

<sup>1</sup> Lista przetestowanych wersji przeglądarek internetowych znajduje się w rozdziale 5.

## 2. TEST DZIAŁANIA SIDGETÓW

### 2.1. WSTĘP

W celu sprawdzenia prawidłowości połączenia komputera z urządzeniem **MultiCon** i poprawności działania silnika sidgetów utworzymy wizualizację testową. Zostanie ona przygotowana lokalnie (na komputerze użytkownika) jako najprostsza strona WWW.

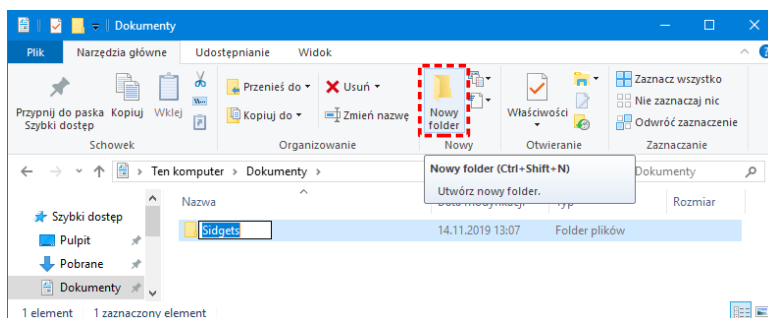
### 2.2. PRZYGOTOWANIE INFRASTRUKTURY

Wizualizacje z sidgetami działają dzięki stałemu pobieraniu za pomocą połączenia sieciowego przez przeglądarkę Internetową danych z urządzeń **MultiCon**. Dlatego też, do przeprowadzenia testu działania sidgetów, niezbędne będzie przygotowanie następujących składników:

- urządzenie **MultiCon** z oprogramowaniem firmware w wersji, co najmniej **5.07.0** podłączone do sieci LAN na znanym adresie IP, np. 192.168.1.222
- komputer klasy PC użytkownika podłączony to tej samej sieci LAN, co powyższe urządzenie (możliwe jest także połączenie bezpośrednie z urządzeniem)
- komputer z dowolnym systemem operacyjnym Windows (kurs dotyczy Windows 10)

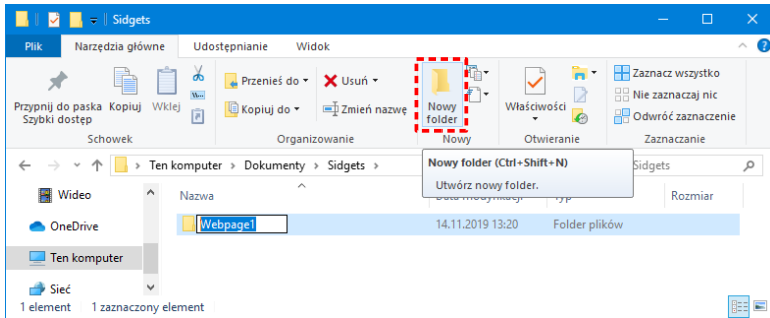
### 2.3. PRZYGOTOWANIE STRONY TESTOWEJ

W celu utworzenia na komputerze użytkownika najprostszej strony WWW wykorzystującej silnik sidgetów, zaleca się stworzenie specjalnego folderu, w którym będziemy umieszczać tego typu wizualizacje. W tym celu otworzymy **Eksploreator plików** i w wybranej lokalizacji komputera stwórzmy folder o dowolnej nazwie, np. „Sidgets”. Sposób tworzenia takiego folderu na systemie Windows 10 przedstawia rysunek poniżej.



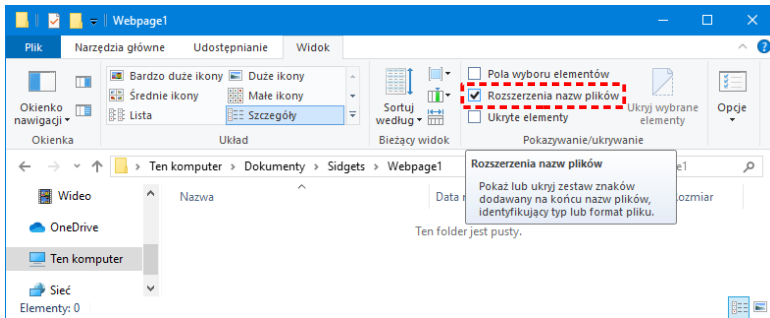
Rys. 2.1 Tworzenie folderu grupującego różne wizualizacje

Dobłą praktyką jest tworzenie dodatkowo w powyższym folderze odrębnych podfolderów dla wizualizacji wykorzystujących wspólne zasoby, np. grafiki, style, itp. Na nasze potrzeby stwórzmy więc dodatkowy podfolder o nazwie „Webpage1” (Rys. 2.2).



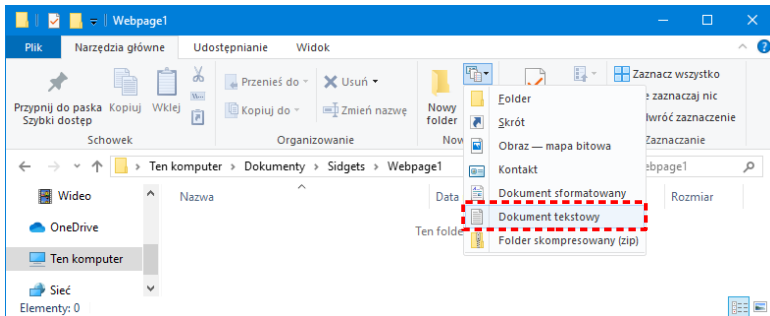
Rys. 2.2 Tworzenie folderu dla pierwszej wizualizacji

Przed utworzeniem pliku z pierwszą wizualizacją należy upewnić się, że wyświetlanie rozszerzeń nazw plików jest włączone (Rys. 2.3).



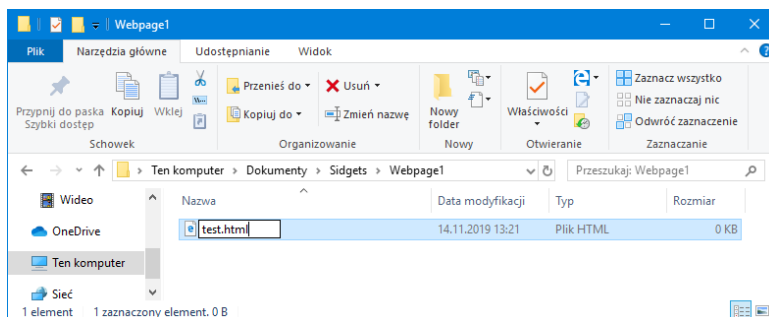
Rys. 2.3 Włączona opcja „Rozszerzenia nazw plików”

W celu utworzenia testowej strony wizualizacyjnej, należy w folderze „Webpage1” utworzyć nowy plik tekstowy za pomocą odpowiedniej opcji w menu **Eksploratora plików** lub menu kontekstowego myszy (Rys. 2.4).



Rys. 2.4 Tworzenie nowego pliku strony Web

W trakcie tworzenia pliku tekstowego należy nadać mu odpowiednią nazwę, zmieniając domyślne rozszerzenie \*.txt na \*.html (Rys. 2.5).



Rys. 2.5 Nadanie właściwej nazwy stronie Web

W ten sposób stworzono pusty plik strony sieci Web. Należy teraz uzupełnić jego zawartość podstawowym kodem HTML – niezbędnym do poprawnego uruchomienia wizualizacji w przeglądarce internetowej.

Do tego celu można użyć systemowego narzędzia **Notatnik**. Aby to zrobić, należy otworzyć menu kontekstowe myszy na wcześniej utworzonym pliku i wybrać kolejno polecenia:

**"Otwórz za pomocą" > "Wybierz inną aplikację" > "Więcej aplikacji ↑" > "Notatnik" > OK**

Po otwarciu pliku „test.html” w **Notatniku**, należy wpisać lub wkleić do niego poniższy kod HTML.

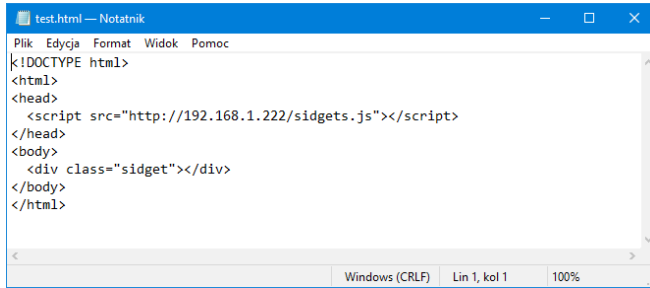
```

1 <!DOCTYPE html>
2 <html>
3 <head>
4   <script src="http://192.168.1.222/sidgets.js"></script>
5 </head>
6 <body>
7   <div class="sidget"></div>
8 </body>
9 </html>

```

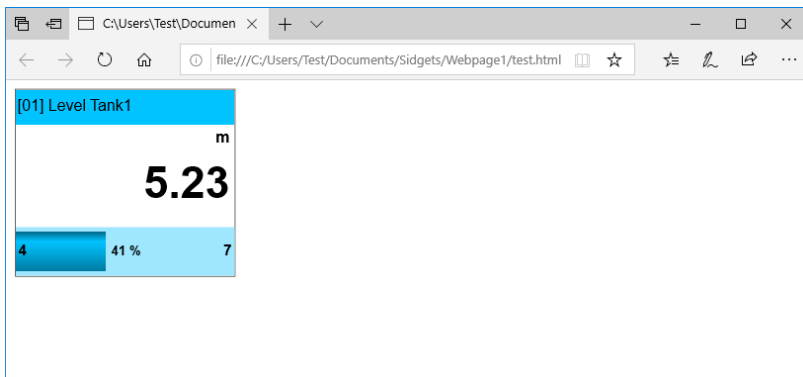
Listing 2.1. Kod HTML testujący działanie silnika sidgetów

W efekcie powyższych działań, okno **Notatnika** powinno wyglądać jak na poniższym rysunku.



Rys. 2.6 Okno **Notatnika** z wklejonym kodem HTML

Po zapisaniu zmian w pliku „test.html”, można przetestować jego działanie. W tym celu wystarczy uruchomić ten plik poprzez podwójne kliknięcie z poziomu **Ekspłoratora plików**. Wizualizacja zostanie uruchomiona za pomocą domyślnej przeglądarki internetowej przypisanej do obsługi plików z rozszerzeniami \*.html. Jeśli połączenie z urządzeniem działa prawidłowo, to po krótkiej chwili powinien na ekranie pojawić się obiekt graficzny wyświetlający nie tylko bieżący pomiar, ale także aktualne ustawienia jednego z kanałów pomiarowych z jakimi działa urządzenie **MultiCon** (Rys. 2.7).



Rys. 2.7 Wygląd wizualizacji testowej w przeglądarce **Microsoft Edge**

Wyjaśnienie zasady działania użytego kodu HTML jest opisane w rozdziale 4.

## **3. PRZYGOTOWANIE ŚRODOWISKA DO TWORZENIA WIZUALIZACJI**

### **3.1. WSTĘP**

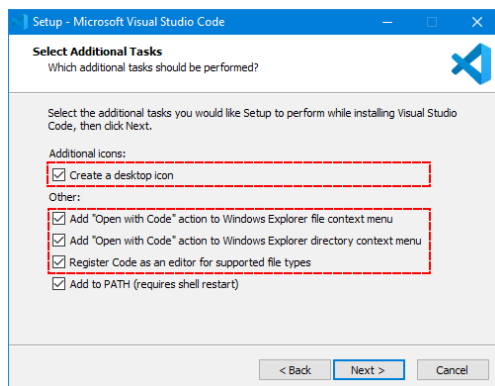
Choć tworzenie i testowanie stron sieci Web za pomocą narzędzi dostarczanych wraz z systemem operacyjnym Windows (**Eksplorator plików**, **Notatnik**, przeglądarka **Microsoft Edge**) jest możliwe i wystarczające do utworzenia strony w celach testowych, to do tworzenia już bardziej rozbudowanych wizualizacji zaleca się zainstalowanie odpowiednich narzędzi, które w sposób naturalny wspomogą ten proces.

W niniejszym rozdziale zostanie przedstawiony sposób przygotowania darmowego środowiska edytorskiego, które w sposób szybki i przyjazny użytkownikowi umożliwi tworzenie wielu stron wizualizacyjnych wykorzystujących silnika sidgetów. Nie ograniczy jednocześnie możliwości edycji kodu stron Web użytkownikom zajmującym się na co dzień profesjonalnie tymi zagadnieniami.

Jednym z edytorów wspomagających tworzenie kodu w technologii HTML, CSS, JavaScript i wielu innych powiązanych z siecią Web, jest darmowy (także do celów komercyjnych) produkt firmy Microsoft – **Visual Studio Code**. Choć niniejszy przewodnik dotyczy tworzenia stron wizualizacyjnych w systemie operacyjnym Windows, nic nie stoi na przeszkodzie, by odpowiednie działania zastosować w innych systemach operacyjnych. Edytor **Visual Studio Code** dostępny jest także na platformy Linux i macOS.

### **3.2. INSTALACJA VISUAL STUDIO CODE**

Po pobraniu ze strony <https://code.visualstudio.com/> instalatora programu **Visual Studio Code**, należy go zainstalować. Proces instalacji przebiega w sposób typowy dla programów systemu Windows, warto jednak zwrócić uwagę na etap, w którym istnieje możliwość wykonania dodatkowych działań. Najistotniejsze z nich ukazano na poniższym rysunku.



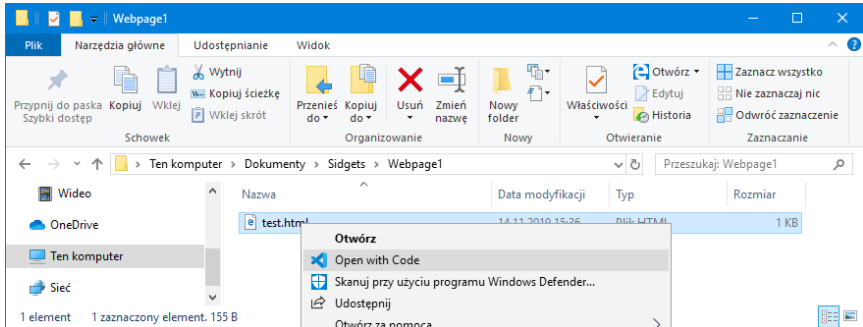
*Rys. 3.1 Zaznaczenie dodatkowych opcji instalacyjnych, które ułatwią późniejszą pracę z programem*

Zainstalowanie programu zgodnie z sugestiami z rysunku ułatwi użytkownikowi wykonywanie kroków w sposób przedstawiony w niniejszym kursie.



### 3.3. URUCHOMIENIE VISUAL STUDIO CODE

Gdy program **Visual Studio Code** jest już zainstalowany, można go uruchomić za pomocą utworzonego w systemie operacyjnym skrótu. Mając jednak wcześniej już przygotowaną stronę testową „test.html” (patrz Rozdział 2), można wykorzystać nowe polecenie **Open with Code**, które pojawiło się w menu kontekstowym plików \*.html (Rys. 3.2). Polecenie to umożliwi natychmiastową edycję tego pliku za pomocą nowego edytora.

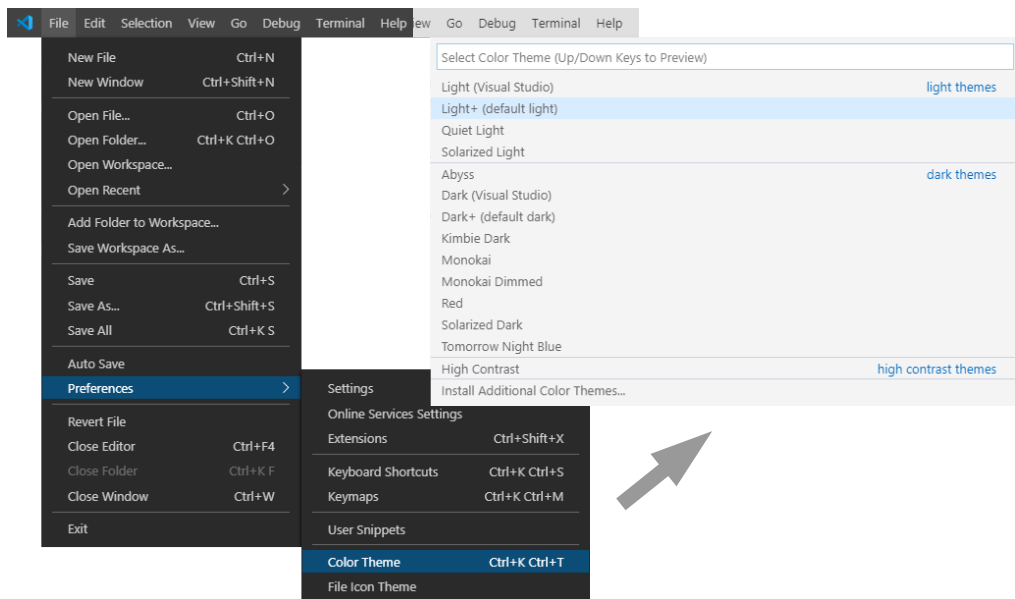


Rys. 3.2 Edycja pliku \*.html za pomocą **Visual Studio Code**

Powyższy sposób otwierania plików jest zbliżony do stosowanego przy edycji pliku za pomocą systemowego **Notatnika**, jednak dla bardziej złożonych projektów poleca się otwieranie całych folderów bezpośrednio już z samego programu.

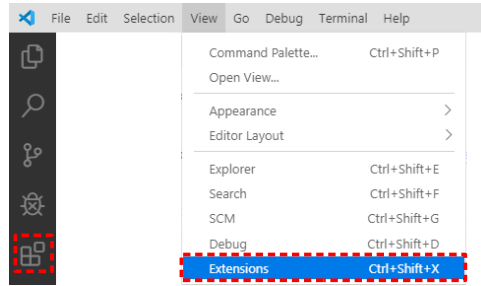
### 3.4. DOSTOSOWANIE VISUAL STUDIO CODE

Kolorystyka wyświetlania interfejsu użytkownika i kodu w programie **Visual Studio Code** jest domyślnie ustawiona na tryb ciemny. Niniejszy przewodnik używa zrzutów ekranu oraz składni kodu w kolorystyce jasnej. W celu zachowania spójności, użytkownik może również zmienić ten tryb wybierając odpowiednie polecenie z menu **File > Preferences > Color Theme** lub naciskając skrót klawiaturowy **Ctrl + (K > T)**, a następnie wybierając pozycję „Light+ (default light)” (Rys. 3.3).



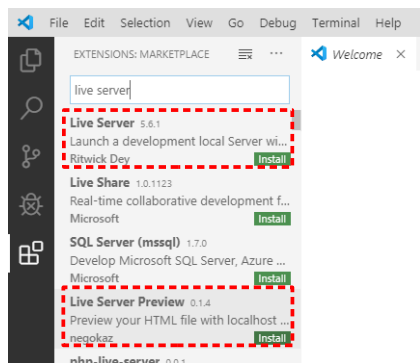
Rys. 3.3 Zmiana schematu kolorów na jasny

Kolejną czynnością jaką warto wykonać w środowisku **Visual Studio Code** na potrzeby pracy z sidgetami, jest doinstalowanie rozszerzeń ułatwiających szybki podgląd efektu końcowego edytowanej strony. W tym celu należy otworzyć menedżer rozszerzeń za pomocą polecenia **Extensions** z menu **View** lub za pomocą kliknięcia na odpowiedniej ikonie w menu bocznym (Rys. 3.4).



Rys. 3.4 Otwieranie menedżera rozszerzeń

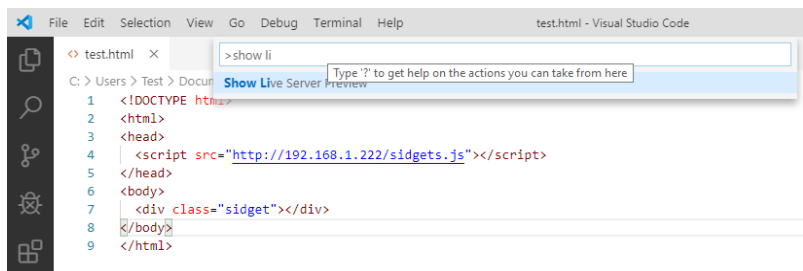
Następnie, należy wpisać w polu wyszukiwania frazę „live server”. Ograniczy to listę do takich pozycji, wśród których znajdują się interesujące nas rozszerzenia. Rozszerzenia, które warto zainstalować to **Live Server Preview** i **Live Server**. Aby tego dokonać, należy kliknąć na przycisku **[Install]** przy każdym z nich (Rys. 3.5).



Rys. 3.5 Rozszerzenia zalecane do instalacji

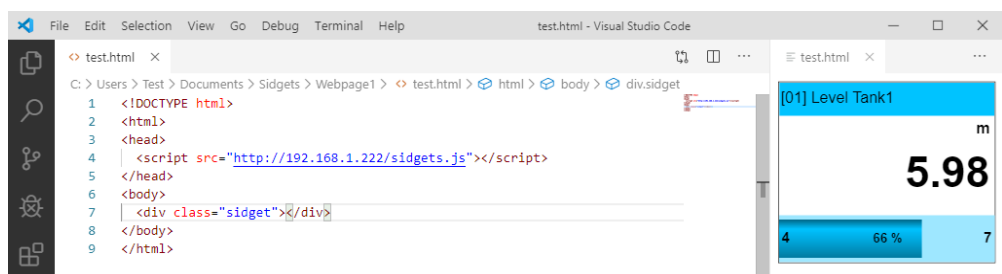
Rozszerzenie **Live Server Preview** służy do szybkiego podglądu efektu końcowego tworzonej strony z kodem HTML w taki sposób, w jaki strona renderowana by była w przeglądarce internetowej.

W celu przetestowania działania rozszerzenia, należy się upewnić, że w programie jest otwarty do edycji dowolny plik \*.html, np. wcześniej utworzony plik „test.html”. Następnie należy wybrać polecenie **Command Palette...** z menu **View** lub nacisnąć skrót klawiaturowy **F1** i po odszukaniu, uruchomić polecenie „Show Live Server Preview” (Rys. 3.6).



Rys. 3.6 Wybór polecenia uruchamiającego podgląd bieżącego pliku

Spowoduje to uruchomienie osobnej zakładki programu, w której zostanie wyświetlony podgląd działania właśnie modyfikowanego pliku \*.html (Rys. 3.7).

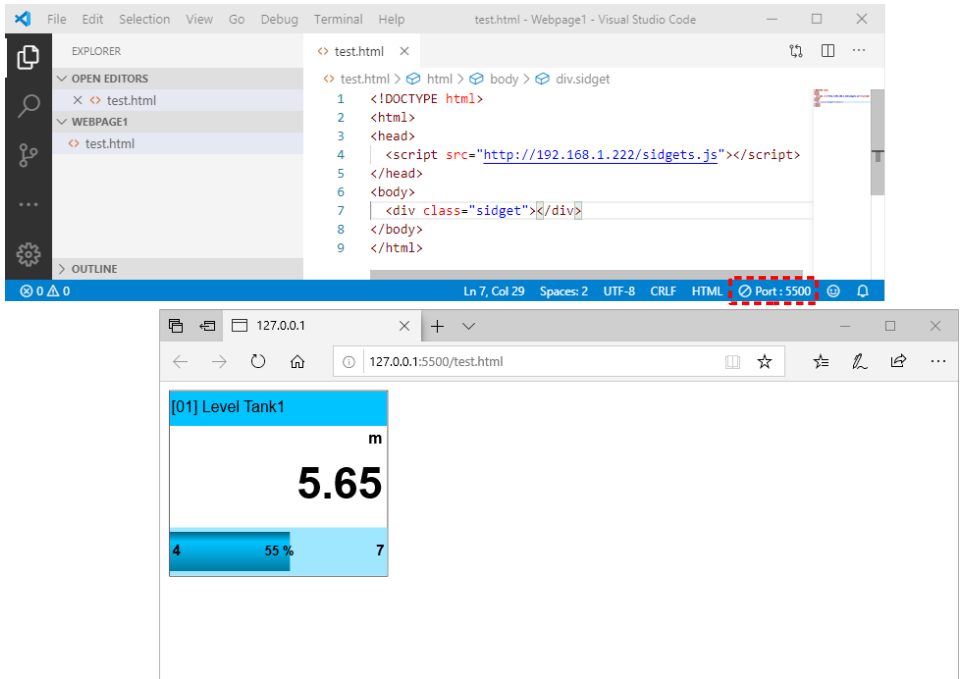


Rys. 3.7 Podgląd działania strony Web dzięki rozszerzeniu Live Server Preview

W trakcie dalszej rozbudowy strony sieci Web z sidgetami, każde zapisanie zmian w pliku \*.html powodować będzie automatyczne odświeżenie zawartości zakładki podglądu. Rozwiązanie takie jest praktyczniejsze, niż częste przełączanie się pomiędzy środowiskiem Visual Studio Code, a przeglądarką internetową, której zawartość dodatkowo należałoby odświeżać.

Dla bardziej rozbudowanych stron www, łatwego dostępu do całego obszaru ich zawartości lub większej kontroli za pomocą narzędzi developerskich przeglądarki nad wykonywanym kodem, poleca się skorzystać z udogodnień oferowanych przez rozszerzenie **Live Server**. Rozszerzenie to, podobnie jak **Live Server Preview**, śledzi zapisywanie zmian w pliku strony i automatycznie odświeża jej podgląd. Działa to już jednak w ramach okna odrębnej przeglądarki internetowej, na której chcemy testować naszą stronę.

W celu przetestowania działania rozszerzenia **Live Server**, nie wystarczy już tylko, jak w przypadku rozszerzenia **Live Server Preview**, otworzyć do edycji pojedynczy plik \*.html, lecz wymagane jest otwarcie całego projektu z tworzoną stroną www. Sprowadza się to zwykle do otwarcia folderu nadrzędnego, w którym nasza strona jest umieszczona. Otwórzmy więc wcześniej utworzony w rozdziale 2.3 folder „Webpage1” za pomocą polecenia **Open Folder...** z menu **File**. Spowoduje to pojawienie się w menu bocznym programu struktury plików tego folderu. Następnie z tak wyświetlonej struktury należy otworzyć do edycji plik „test.html”. Kliknięcie na polu **Go Live** na pasku statusu spowoduje uruchomienie uproszczonej wersji lokalnego serwera www i otwarcie projektowanej strony wizualizacyjnej w oknie domyślnej przeglądarki internetowej. W czasie, gdy serwer www jest aktywny, pole **Go Live** powinno zmienić nazwę na taką, która wskazuje numer portu na jakim działa serwer (Rys. 3.8).



Rys. 3.8 Podgląd działania strony Web dzięki rozszerzeniu **Live Server**

Dopóki serwer www jest w trakcie działania, to adres URL z domyślnie otwartej przeglądarki internetowej może zostać również skopiowany i uruchomiony w innej przeglądarce zainstalowanej na komputerze.

Konfiguracja środowiska **Visual Studio Code** zgodnie z powyższymi wytycznymi nie jest konieczna i nie musi się ograniczać tylko do tych zmian, jednak stanowi ona punkt wyjściowy dla wygodnego tworzenia stron www z sidgetami.

## 4. TWORZENIE WIZUALIZACJI

### 4.1. WSTĘP

Tworzenie stron wizualizacyjnych z zastosowaniem sidgetów polega na deklaracji wytycznych dla poszczególnych obiektów, które to określają wygląd, zachowanie i źródło danych. Cała logika dbająca o spełnienie tych deklaracji wykonuje się automatycznie. W rozdziale tym zostanie omówiony sposób działania wszystkich deklaracji, które użytkownik może określić podczas tworzenia wizualizacji.

Po ukończeniu niniejszego rozdziału, użytkownik będzie posiadał umiejętności, które pozwolą mu na rozumienie zasady działania i tworzenia prostych wizualizacji z wykorzystaniem sidgetów. Pogłębienie wiedzy z zakresu tworzenia stron www może dodatkowo rozszerzyć możliwości, o których w tym przewodniku nie wspomniano.

### 4.2. PODSTAWOWA STRUKTURA WIZUALIZACJI

Każda poprawna strona internetowa powinna zawierać kod o strukturze, co najmniej przedstawionej na listingu 4.1.

```

1 <!DOCTYPE html>
2 <html>
3 <head>
4
5 </head>
6 <body>
7
8 </body>
9 </html>

```

*Listing 4.1. Podstawowa struktura każdej strony www*

Wynika to z określonych wytycznych dotyczących budowy stron internetowych z wykorzystaniem języka HTML<sup>2</sup>.

W podpunkcie 2.3 przedstawiono sposób utworzenia prostej wizualizacji testowej, która wyświetla pojedynczy sidget. Do tego celu posłużono się najbardziej podstawowym kodem HTML, który powinien znajdować się w każdej wizualizacji wykorzystującej sidgety. Na poniższym listingu przedstawiono go ponownie.

```

1 <!DOCTYPE html>
2 <html>
3 <head>
4   <script src="http://192.168.1.222/sidgets.js"></script>
5 </head>
6 <body>
7   <div class="sidget"></div>
8 </body>
9 </html>

```

*Listing 4.2. Kod HTML testujący działanie silnika sidgetów*

Jak można zauważyć, poza podstawową strukturą dokumentu HTML, w liniach 4 i 7 został zadeklarowany dodatkowy kod. Są to deklaracje związane technologią sidgetów.

<sup>2</sup> Szerszą wiedzę na temat budowy stron internetowych w technologii HTML można uzyskać z wielu powszechnie dostępnych książek i kursów

Linia

```
4 <script src="http://192.168.1.222/sidgets.js"></script>
```

powoduje dołączenie do strony WWW silnika odpowiedzialnego za automatyczne tworzenie sidgetów w zadeklarowanych przez użytkownika miejscach strony oraz odświeżanie ich pomiarów.



*Silnik „sidgets.js” znajduje się w każdym urządzeniu **MultiCon** z wersją firmware co najmniej **5.07**. Jeśli urządzenie posiada niższą wersję firmware, to chcąc skorzystać z funkcjonalności tworzenia własnych wizualizacji z wykorzystaniem tej technologii, należy najpierw zaktualizować oprogramowanie urządzenia do najnowszej dostępnej wersji.*

Kolejna linia

```
7 <div class="sidget"></div>
```

informuje silnik, że powyższy znacznik `<div>` będzie stanowił kontener dla sidgetu. Dzieje się tak dlatego, że do elementu przypisano klasę `sidget`, a wszystkie znaczniki HTML działające jako kontenery blokowe, np. `<div>`, lub jako kontenery liniowe, np. `<span>`, które tę klasę posiadają stanowią miejsce, wewnątrz którego zostanie automatycznie utworzony sidget. Tagi takie mogą być umieszczone w dowolnym miejscu w strukturze dokumentu HTML.

Kilka innych poprawnych tagów, wewnątrz których zostaną utworzone sidgety:

```
<span class="sidget"></span>
<p class="sidget"></p>
<object class="sidget"></object>
<table>
  <tr>
    <td class="sidget"></td>
    <td class="sidget"></td>
  </tr>
</table>
```

### 4.3. ATRYBUTY SILNIKA SIDGETÓW

Silnik sidgetów może zostać uruchomiony ze specjalnymi parametrami zmieniającymi sposób jego działania. Parametry te mogą być przydatne podczas tworzenia i testowania wizualizacji. Deklaruje się je w postaci atrybutów z przedrostkiem `data-...` w znaczniku `<script>`, np.

```
<script src="http://192.168.1.5/sidgets.js" data-mode=1 data-demo="random"></script>
```

Lista możliwych do zadeklarowania atrybutów silnika sidgetów jest następująca:

**data-mode** – określa tryb, w jakim silnik uruchomi wizualizację

#### Możliwe wartości

- 0 **tryb podglądu** (domyślny) – stanowi podstawowy tryb działania ukończonej wizualizacji
- 1 **tryb edycji** – odblokowuje funkcjonalność zmiany pozycji sidgetów i uruchamia narzędzia do identyfikacji ich ustawień

Określenie trybu działania silnika nie jest konieczne do tworzenia wizualizacji, jednak uruchomienie **trybu edycji** znacznie ułatwia ten proces. Sposób działania tego ustawienia jest omówiony w rozdziale **4.6 TWORZENIE WIZUALIZACJI W PRAKTYCE**.



*Należy zwrócić szczególną uwagę, by przed oddaniem wizualizacji do użytku usunąć ten parametr lub ustawić jego wartość na 0.*

**data-demo** – określa sposób pozyskiwania wartości pomiarowej przez sidgety

#### Możliwe wartości

- "chan" (domyślny) – każdy sidget wyświetla zmieniającą **wartość rzeczywistą pomiaru** pobieraną na bieżąco z wybranego kanału urządzenia
- "random" – każdy sidget wyświetla zmieniającą się **wartość symulowaną wygenerowaną losowo** z zakresu nieco wykraczającego poza górne i dolne ograniczenie ustawione w kanale urządzenia
- "wartość" – każdy sidget wyświetla tę samą **wartość stałą** podaną w tym atrybucie, np. "-5.3", "912", "-inf", "inf"

Możliwość zasymulowania przez sam silnik zmieniających się wartości pomiarowych może ułatwić tworzenie różnych wizualizacji, nawet bez ustawiania docelowej logiki w urządzeniu **MultiCon**. Należy jednak pamiętać, że mimo symulacji samego pomiaru, ustawienia kanałów takie jak nazwa, jednostka, precyzja wyświetlania, granice wykresu, itd. są nadal pobierane bezpośrednio z urządzenia i wyświetlane w sidgetach.



*Należy zwrócić szczególną uwagę, by przed oddaniem wizualizacji do użytku usunąć ten parametr.*

**data-hide-inactive** – określa sposób sterowania widocznością kontenerów z sidgetami dla nieaktywnych kanałów

#### Możliwe wartości

- 0 **nie ukrywaj** (domyślny) – wszystkie kontenery z sidgetami są widoczne
- 1 **ukrywaj, gdy kanał nieaktywny** – ukrywane są kontenery z sidgetami, w których zadeklarowany kanał nie istnieje lub jest nieaktywny





Ukryty kontener z sidgetem w wyniku działania atrybutu `data-hide-inactive=1` nie zajmuje miejsca na stronie. Warto więc wykorzystać ten atrybut do projektów bardziej uniwersalnych stron z automatycznie dostosowywanym układem uzależnionym od włączonych kanałów.

#### 4.4. PARAMETRY SIDGETÓW

Przykład z listingu 4.2 działa prawidłowo. Trudno jednak znaleźć jego praktyczne zastosowanie, bo jak można zauważyć, wyświetla się pomiar z kanału numer 1 w postaci wskaźnika numerycznego, mimo że tego jawnie nie określono. Są to wartości domyślne każdego sidgetu, które rzadko odpowiadają rzeczywistym potrzebom. Dlatego też w każdym kontenerze należy podać dodatkowe parametry precyzujące źródło danych i sposób ich prezentacji.

Parametry, które można określić dla każdego sidgetu lub ich grupy, należy podawać w postaci stylu CSS. Przykładowo chcąc określić, że wartość pomiaru ma być pobierana z kanału numer 2, należy zmodyfikować nieco linię deklaracji obiektu do postaci:

```
7 <div class="sidget" style="--sid-chan:2;"></div>
```

Jak można zauważyć, parametr zaczyna się od przedrostka `--sid-....`. Jest to cecha wspólna wszystkich parametrów specyficznych dla obiektów typu sidget i pozwala łatwo odróżnić je od właściwości stylów zawartych już w samym standardzie języka CSS.

Parametry określające ustawienia sidgetów są następujące:

`--sid-host` – określa urządzenie źródłowe, z którego będą odczytywane pomiary

Dozwolone wartości	Wartość domyślna	Przykładowe wartości
numer IP lub nazwa hosta urządzenia <b>MultiCon</b> (może zawierać numer portu HTTP)	host podany w <code>src</code> w znaczniku <code>&lt;script&gt;</code>	'192.168.1.151:80' 'dev1.multicon.com' '192.168.100.5'

Sidgety umożliwiają odczyt na jednej stronie sieci Web danych pomiarowych pochodzących z wielu urządzeń jednocześnie. Jeśli określimy za pomocą parametru `--sid-host` inne urządzenie niż to, z którego jest pobierany silnik „sidgets.js”, to sidget lub ich grupa, których dotyczy ten parametr, będą pobierały pomiary właśnie z tego źródła danych.

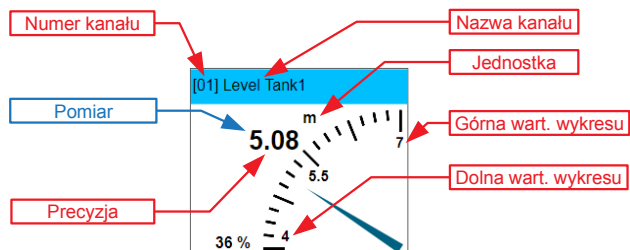


*Maksymalna liczba możliwych źródeł danych dla jednej strony www jest ograniczona przez wykorzystywaną przeglądarkę internetową. W celu zachowania pełnej kompatybilności między różnymi przeglądarkami, zaleca się nie przekraczać liczby 5 różnych hostów dla jednej wizualizacji.*

`--sid-chan` – określa numer kanału, z którego będą pobierane ustawienia i pomiary

Dozwolone wartości	Wartość domyślna	Przykładowe wartości
1 ... 60 (MultiCon 99/N16) lub 1 ... 90 (MultiCon 141)	1	15 3

Parametr ten określa numer kanału urządzenia, z którego będą pobierane ustawienia i pomiary bieżące. Pozyskane w ten sposób informacje zostaną automatycznie wykorzystane przez wybrany sidget (Rys. 4.1).

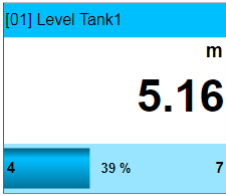
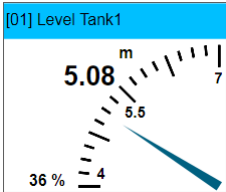


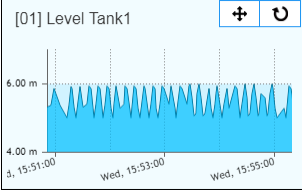
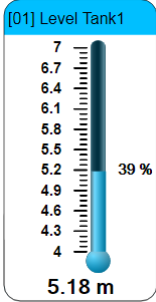



Rys. 4.1 Przykład wyświetlania przez sidget pomiaru i ustawień dla wybranego kanału

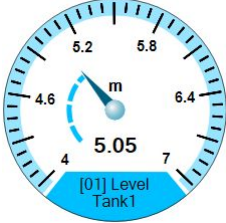
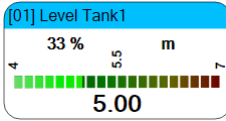
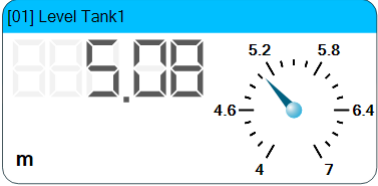
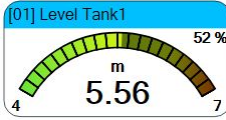
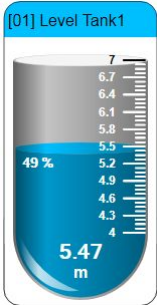
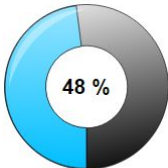
--sid-type – określa typ użytego sidgetu

Dozwolone wartości	Wartość domyślna	Przykładowe wartości
0 ... 13 lub 'nazwa'	1	2 'Analog Meter' 'analogmeter'

Parametr ten określa typ sidgetu. Jest on związany z odmiennym wyglądem i zachowaniem obiektu, w stosunku do innych typów. Jako wartość tego parametru można podać **Numer** lub **Nazwę** spośród dopuszczalnych wartości przedstawionych w poniższej tabeli.

Numer	Nazwa	Wygląd
0	Text	<i>Bieżąca wartość kanału „Level Tank1” wynosi 5.16 m i stanowi 39% zakresu</i>
1 (domyślny)	Value	
2	Needle	

Numer	Nazwa	Wygląd								
3	Graph									
4	Thermometer									
5	Two-state LED	<p data-bbox="748 703 904 735">  [01] Level Tank1         </p> <table border="1" data-bbox="591 762 1065 884"> <thead> <tr> <th>Pomiar</th> <th>Kolor LED</th> </tr> </thead> <tbody> <tr> <td><math>P \leq 0</math></td> <td>(przezroczysty)</td> </tr> <tr> <td><math>P &gt; 0</math></td> <td>--sid-color</td> </tr> </tbody> </table>	Pomiar	Kolor LED	$P \leq 0$	(przezroczysty)	$P > 0$	--sid-color		
Pomiar	Kolor LED									
$P \leq 0$	(przezroczysty)									
$P > 0$	--sid-color									
6	Three-state LED	<p data-bbox="748 906 904 938">  [01] Level Tank1         </p> <table border="1" data-bbox="591 965 1065 1125"> <thead> <tr> <th>Pomiar</th> <th>Kolor LED</th> </tr> </thead> <tbody> <tr> <td><math>P &gt;</math> Wart. górna wykresu</td> <td>--sid-hicolor</td> </tr> <tr> <td>dowolny</td> <td>--sid-color</td> </tr> <tr> <td><math>P &lt;</math> Wart. dolna wykresu</td> <td>--sid-locolor</td> </tr> </tbody> </table>	Pomiar	Kolor LED	$P >$ Wart. górna wykresu	--sid-hicolor	dowolny	--sid-color	$P <$ Wart. dolna wykresu	--sid-locolor
Pomiar	Kolor LED									
$P >$ Wart. górna wykresu	--sid-hicolor									
dowolny	--sid-color									
$P <$ Wart. dolna wykresu	--sid-locolor									
7	Three-state Rectangular LED	<p data-bbox="748 1147 904 1179">  [01] Level Tank1         </p> <table border="1" data-bbox="591 1190 1065 1343"> <thead> <tr> <th>Pomiar</th> <th>Kolor LED</th> </tr> </thead> <tbody> <tr> <td><math>P &gt;</math> Wart. górna wykresu</td> <td>--sid-hicolor</td> </tr> <tr> <td>dowolny</td> <td>--sid-color</td> </tr> <tr> <td><math>P &lt;</math> Wart. dolna wykresu</td> <td>--sid-locolor</td> </tr> </tbody> </table>	Pomiar	Kolor LED	$P >$ Wart. górna wykresu	--sid-hicolor	dowolny	--sid-color	$P <$ Wart. dolna wykresu	--sid-locolor
Pomiar	Kolor LED									
$P >$ Wart. górna wykresu	--sid-hicolor									
dowolny	--sid-color									
$P <$ Wart. dolna wykresu	--sid-locolor									

Numer	Nazwa	Wygląd
8	Analog Meter	
9	Horizontal LED Bar	
10	Digital & Analog	
11	Arc LED Bar	
12	Tank	
13	Pie Chart	

**--sid-interval** – określa w sekundach czas pomiędzy odświeżaniem pomiaru

Dozwolone wartości	Wartość domyślna	Przykładowe wartości
1 ... n	1	5 300

Parametr ten określa interwał czasowy wskazujący sidgetowi, jak często powinien odświeżać swoją wartość pomiarową. Czas ten wyrażony jest w pełnych sekundach.



*Wewnętrzny mechanizm silnika „sidgets.js” stale odpytuje wszystkie urządzenia o nowe pomiary z częstotliwością wynikającą z ustawionych przez użytkownika interwałów czasowych. Jeśli kilka sidgetów, które odnoszą się do tego samego urządzenia, ma ustawione inne interwały, to silnik tak zoptymalizuje odpytywanie urządzeń, by zapewnić deklarowane czasy odświeżenia wartości, przy zachowaniu minimalnego obciążenia tego urządzenia. Ustawienia kanałów są natomiast pobierane stale co 15 sekund.*

Ustawienie parametru **--sid-interval** ma szczególne znaczenie dla sidgetów wykorzystujących dane historyczne, np. wykresy (Graph), czy sidgety tekstowe (Text). Każdy taki sidget ma bufor **600 pomiarów**, więc każdy wykres może wyświetlać, co najwyżej tyle punktów z przeszłości. Oznacza to, że dobierając odpowiedni interwał aktualizacji, możemy sterować rozpiętością czasu, do jakiego można sięgnąć w przeszłość, by móc odczytać zarejestrowany pomiar. Przykładowo, określenie czasu aktualizacji jako **--sid-interval:60;** oznacza, że każdy punkt do wykresu będzie dodawany co minutę. Skoro wykres może ich zgromadzić 600, to  $60s \cdot 600 = 36000s = 10g$ . Wykres w takim przypadku będzie przedstawiał dane do 10 godzin wstecz, gdzie każde dodanie nowego punktu, będzie powodowało skasowanie najstarszego.

**--sid-color** – określa kolor wiodący sidgetu

parametr nie dotyczy sidgetów typu: **0-Text**

Dozwolone wartości	Wartość domyślna	Przykładowe wartości
nazwa lub wartość numeryczna koloru	DeepSkyBlue	Yellow #14ff01 rgb(83, 83, 245)

Sidgety posiadają mechanizm stylizacji kolorem wiodącym. Polega on na tym, że projekt sidgetu posiada jeden kolor podstawowy, względem którego istotne elementy sidgetu się automatycznie dostosowują, zachowując zaprojektowane poziomy jasności. Funkcjonalność ta pozwala na łatwe dostosowanie kolorystyki dynamicznych obiektów względem pozostałych elementów graficznych strony. Pozwala też wyróżnić wybrane grupy sidgetów.

Parametr **--sid-color** służy do określenia koloru wiodącego sidgetu. Jako wartość parametr przyjmuje kolor podany w postaci nazwy, kodu szesnastkowego lub w formacie RGB – zgodnie ze standardem CSS. Listę wszystkich nazw kolorów można znaleźć na stronie <https://html-color-codes.info/color-names/>.



*Sidgety wyświetlające pomiar w sposób dwustanowy, np. diody, mogą interpretować ten parametr jako kolor oznaczający ich stan aktywny.*

--sid-hicolor – określa kolor fragmentu sidgetu informującego użytkownika o przekroczeniu przez pomiar ustawionej w kanale urządzenia **Wartości górnej wykresu**

parametr nie dotyczy sidgetów typu:  $\emptyset$ -Text

Dozwolone wartości	Wartość domyślna	Przykładowe wartości
nazwa lub wartość numeryczna koloru	kolor zależny od typu sidgetu, zwykle Red	Magenta #14ff01 rgb(83, 83, 245)

--sid-locolor – określa kolor fragmentu sidgetu informującego użytkownika o przekroczeniu przez pomiar ustawionej w kanale urządzenia **Wartości dolnej wykresu**

parametr nie dotyczy sidgetów typu:  $\emptyset$ -Text

Dozwolone wartości	Wartość domyślna	Przykładowe wartości
nazwa lub wartość numeryczna koloru	kolor zależny od typu sidgetu, zwykle Blue	Green #14ff01 rgb(83, 83, 245)

Większość sidgetów posiada mechanizm informowania o przekroczeniu przez pomiar ustawionego w kanale urządzenia zakresu wyświetlanych wartości za pomocą ustawień **Wartość górna wykresu** i **Wartość dolna wykresu**.

W przypadku, gdy pomiar zwiększy się i przekroczy górne ograniczenie, to na sidgecie zostanie wyświetlona stosowna informacja graficzna w kolorze wstępnie zaprojektowanym dla takiego przekroczenia (najczęściej w kolorze **czzerwonym**).

W przypadku, gdy pomiar spadnie poniżej wartości dolnego ograniczenia, to na sidgecie zostanie wyświetlona stosowna informacja graficzna w kolorze wstępnie zaprojektowanym dla takiego przekroczenia (najczęściej w kolorze **niebieskim**).

Przykłady reakcji niektórych sidgetów na takie zdarzenia przedstawiono na rysunku 4.2.



Rys. 4.2 Przykłady reakcji wybranych sidgetów na przekroczenie przez wartość górnego (po lewej) lub dolnego (po prawej) ograniczenia wyświetlania

Parametry `--sid-hicolor` i `--sid-locolor` pozwalają na zmianę domyślnych kolorów elementów graficznych informujących o przekroczeniach.

Zasady dotyczące formatu wartości dla tego parametru są takie same, jak w przypadku parametru `--sid-color`.

`--sid-scaling` – określa w procentach współczynnik skalowania sidgetu względem wymiarów domyślnych

parametr nie dotyczy sidgetów typu: `θ-Text`

Dozwolone wartości	Wartość domyślna	Przykładowe wartości
1, 2, ..., n	100	80 150

Każdy sidget jest zaprojektowany tak, by w skali 1:1 wszystkie jego elementy składowe były możliwie małe, nie rezygnując przy tym z zachowania dobrej czytelności. W zależności od potrzeb, użytkownik może pomniejszyć lub powiększyć taki obiekt zachowując przy tym oryginalne proporcje. Do zmiany skali obiektu służy powyższy parametr, przy czym liczba podana jako wartość, określa wielkość procentową wymiarów domyślnych sidgetu.

Użycie tego parametru nadpisuje możliwość określania wymiarów za pomocą właściwości `width` i `height` w stylu sidgetu.

#### **4.5. SPOSOBY DEKLARACJI PARAMETRÓW SIDGETÓW**

W rozdziale 4.4 podano sposób zadeklarowania parametru sidgetu `--sid-chan` bezpośrednio w linii kodu, w którym ten sidget zadeklarowano. Taki sposób deklaracji parametru jest tylko jednym z wielu, z jakich użytkownik może skorzystać. Jako, że parametry sidgetów stanowią właściwości języka CSS (*ang. Cascading Style Sheets*), to można je łączyć z innymi znanymi właściwościami tego języka, np. z właściwością `width`, ostatecznie budując z nich styl wykorzystywany przez pojedynczy lub przez wiele znaczników HTML.

Język CSS definiuje trzy podstawowe możliwości określenia stylu:

- A) styl lokalny, tzw „inline”** – polega na bezpośrednim przypisaniu stylu do elementu HTML poprzez atrybut `style`, np.

```
<body>
  <div class="sidget" style="--sid-type:2; --sid-chan:3; width:300px;"></div>
  <div class="sidget" style="--sid-type:2; --sid-chan:4;"></div>
</body>
```

- B) styl wewnętrzny** – jest to arkusz umieszczony wewnątrz znacznika `<style>` w części nagłówkowej strony, np.

```
<head>
  <style>
    .sidget {
      --sid-type: 2;
    }
  </style>
</head>
<body>
  <div class="sidget" style="--sid-chan:3; width:300px;"></div>
  <div class="sidget" style="--sid-chan:4;"></div>
</body>
```

- C) styl zewnętrzny** – polega na umieszczeniu definicji stylów w oddzielnym pliku o rozszerzeniu \*.css, a w dokumencie go używającym umieszczeniu instrukcji importującej w części nagłówkowej strony, np.

```
.sidget {
  --sid-type: 2;
}
```

mysidgets.css

```
<head>
  <link rel="stylesheet" type="text/css" href="mysidgets.css" />
</head>
<body>
  <div class="sidget" style="--sid-chan:3; width:300px;"></div>
  <div class="sidget" style="--sid-chan:4;"></div>
</body>
```

Ze sposobem umieszczania stylów wiąże się pojęcie *kaskadowości*. Definiuje ona hierarchię źródeł stylów. *Kaskadowość* określa, iż w pierwszej kolejności brane są pod uwagę style pochodzące z arkusza zewnętrznego. Te z kolei mogą zostać nadpisane przez style zdefiniowane w arkuszu wewnętrznym. Style zdefiniowane lokalnie („inline”) znajdują się najbliższej opisywanego elementu strony HTML, więc zdefiniowane tam właściwości będą nadpisywały te, które zostały ustawione za pomocą innych metod.

Wykorzystanie zalet kaskadowości w tworzeniu stron z sidgetami stanowi bardzo przydatną funkcjonalność, dzięki której można szybko określić wspólne lub domyślne cechy wszystkich sidgetów za pomocą wewnętrznego lub zewnętrznego arkusza stylów, jak np. typ, kolor, wielkość, interwał, a parametry szczególne już dla konkretnych obiektów, jak numer kanału, czy inny interwał, można określić już za pomocą stylów lokalnych.



*Szersze omówienie języka CSS wykracza poza ramy niniejszego kursu. W celu pełnego wykorzystania możliwości tworzenia wizualizacji z użyciem sidgetów zaleca się przynajmniej w sposób podstawowy zapoznać się ze składnią definiowania stylów i najważniejszymi właściwościami tego języka.*

## **4.6. TWORZENIE WIZUALIZACJI W PRAKTYCE**

Znajomość struktury wizualizacji (rozdział 4.2), znaczenia parametrów sidgetów (rozdział 4.4) i sposobów ich deklaracji (rozdział 4.5) pozwalana na stworzenie pierwszej strony sieci Web na potrzeby rzeczywistej aplikacji.

Na listingu 4.3 przedstawiono kod pliku \*.html, który pobiera silnik sidgetów z urządzenia o adresie IP 192.168.1.100 i deklaruje dwa sidgety. Ponadto każdy ze wskaźników domyślnie powinien wyświetlać się w kolorze żółtym oraz pobierać pomiary i ustawienia z urządzenia o adresie IP 192.168.1.200 i z kanału numer 3. W deklaracji drugiego sidgetu, poprzez wykorzystanie kaskadowości stylów, zmieniono za pomocą stylu lokalnego numer kanału na 5.



```

1 <!DOCTYPE html>
2 <html>
3 <head>
4   <script src="http://192.168.1.100/sidgets.js"></script>
5   <style>
6     .sidget {
7       --sid-host: '192.168.1.200';
8       --sid-color: Yellow;
9       --sid-chan: 3;
10    }
11  </style>
12 </head>
13 <body>
14   <div class="sidget"></div>
15   <div class="sidget" style="--sid-chan:5;"></div>
16 </body>
17 </html>

```

Listing 4.3. Kod HTML pokazujący wykorzystanie możliwości przededefiniowania parametrów

W celu łatwiejszego odróżnienia jednego sidgetu od innego, zaleca się nadanie im unikalnych identyfikatorów za pomocą atrybutu `id`, np.

```

14 <div class="sidget" id="sid1"></div>
15 <div class="sidget" id="tank-level" style="--sid-chan:5;"></div>

```

Łatwiej w ten sposób identyfikować sidgety podczas dalszych modyfikacji wizualizacji, zwłaszcza przy wykorzystaniu **trybu edycji** strony (patrz rozdział 4.3). Włączmy więc ten tryb na czas edycji poprzez dodanie w deklaracji skryptu silnika następującego parametru:

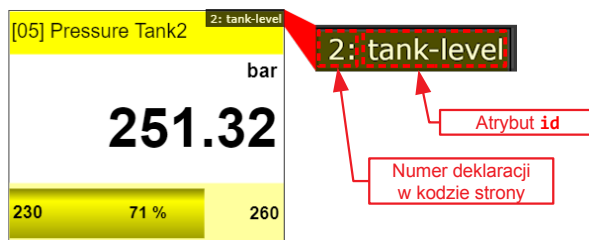
```

4 <script src="http://192.168.1.100/sidgets.js" data-mode=1></script>

```

Strona uruchomiona w tym trybie będzie zawierała dodatkowe ułatwienia i narzędzia przydatne w trakcie jej edycji.

Jednym z takich ułatwień jest pojawienie się dodatkowej etykiety w prawym górnym rogu każdego obiektu, na której jest wyświetlany identyfikator kontenera (atrybut `id`), w którym ten sidget został utworzony. Pojawia się także informacja o numerze tego sidgetu. Numer ten jest związany z kolejnością umieszczenia jego deklaracji w kodzie strony. Informacje te pozwalają na szybkie odnalezienie odpowiedniego zapisu w kodzie, widząc jedynie samą wizualizację (Rys. 4.3).



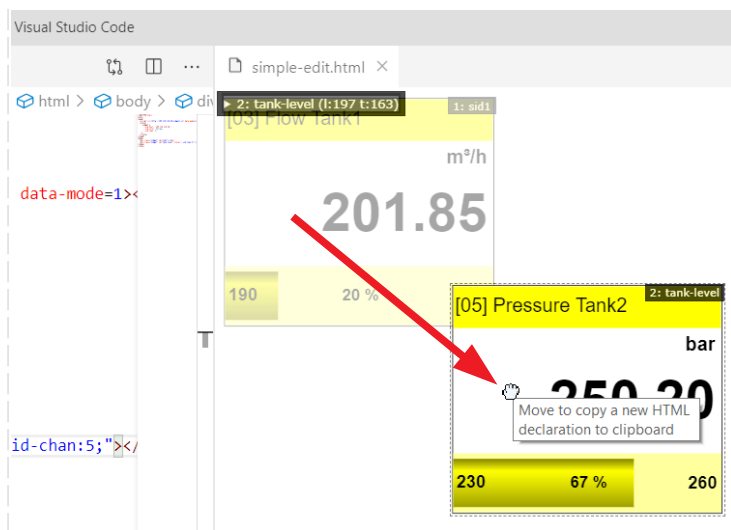
Rys. 4.3 Sidget w trybie edycji z dodatkową etykietą identyfikacyjną

W **trybie edycji**, oprócz etykiety identyfikacyjnej umieszczonej na każdym z sidgetów, w lewym górnym rogu strony pojawia się belka informacyjna. Jest to element, który dostarcza informacji o aktualnie wskazywanym sidgecie. Oprócz powtórzenia informacji z etykiety identyfikacyjnej, dostarcza współrzędne położenia lewego górnego rogu sidgetu. Kliknięcie na belce informacyjnej spowoduje jej rozwinięcie i pokazanie szczegółowych ustawień wskazanego obiektu (Rys. 4.4).



Rys. 4.4 Wyświetlenie informacji dodatkowych o sidgecie

Innym ułatwieniem, które pojawia się w trybie edycji, jest wspomaganie pozycjonowania sidgetów na stronie za pomocą myszy (Rys. 4.5).



Rys. 4.5 Zmiana położenia sidgetu w trybie edycji za pomocą myszy

Po chwyceniu i przeniesieniu za pomocą myszy takiego obiektu w inne miejsce, zostanie skopiowana do schowka właściwa dla tego sidgetu, ale zmodyfikowana już o nową pozycję, deklaracja HTML. Taką skopiowaną do schowka zmodyfikowaną linią, możemy teraz zastąpić oryginalną linię w kodzie strony.

```
15 <div class="sidget" id="tank-level" style="--sid-chan:5; left:197px; top:163px;"></div>
```

Wszystkie elementy klasy `sidget` mają domyślnie ustawiony atrybut `position:absolute`; więc ich pozycja jest określona względem najbliższego znacznika HTML będącego jego rodzicem, którego styl ma właściwość ustawioną na `position:relative`. W związku z tym, poleca się umieszczać sidgety w dodatkowym kontenerze. Utwórzmy więc kontener jako znacznik `<div>` i nazwijmy go `main-container`.

```

14 <div id="main-container" style="position: relative;">
15   <div class="sidget" id="sid1"></div>
16   <div class="sidget" id="tank-level" style="--sid-chan:5; left:197px;
      top:163px;"></div>
17 </div>

```

Taki sposób grupowania sidgetów umożliwi łatwe dodawanie do tej grupy innych elementów strony, względem których pozycja sidgetów musi być zachowana. Może to być np. obrazek tła przechowywany na komputerze użytkownika lub umieszczony w sieci Internet. W celu dodania takiego obrazka wystarczy skorzystać ze znacznika `<img>` i umieścić go we wnętrzu górnej części kontenera.

```

14 <div id="main-container" style="position: relative;">
15   
16   <div class="sidget" id="sid1"></div>
17   <div class="sidget" id="tank-level" style="--sid-chan:5; left:197px;
      top:163px;"></div>
18 </div>

```

Powyższy kod przedstawia sposób odniesienia się do obrazka umieszczonego na komputerze użytkownika w podfolderze „res” znajdującym się w tym samym folderze co strona \*.html.

Kod całej wizualizacji powstałej w wyniku wykonania kroków zgodnie z powyżej przedstawioną koncepcją powinien być podobny do przedstawionego na listingu 4.4.

```

1 <!DOCTYPE html>
2 <html>
3 <head>
4   <script src="http://192.168.1.100/sidgets.js" data-mode=1></script>
5   <style>
6     .sidget {
7       --sid-host: '192.168.1.200';
8       --sid-color: Yellow;
9       --sid-chan: 3;
10    }
11  </style>
12 </head>
13 <body>
14   <div id="main-container" style="position: relative;">
15     
16     <div class="sidget" id="sid1" style="left: 202px; top: 92px;"></div>
17     <div class="sidget" id="tank-level" style="--sid-chan:5;
      left: 50px; top: 505px;"></div>
18   </div>
19 </body>
20 </html>

```

Listing 4.4. Kod HTML wizualizacji w trakcie projektowania

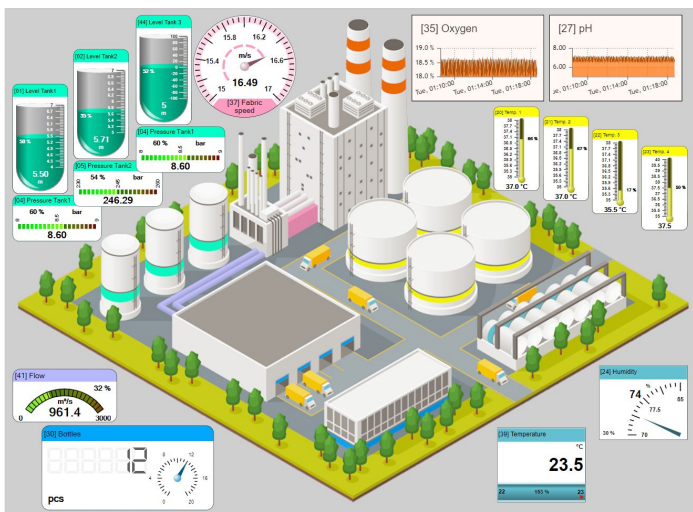
Powyższy kod przekłada się na wizualizację, której wygląd jest przedstawiony na rysunku 4.6.



Rys. 4.6 Wygląd wizualizacji w trakcie projektowania

Powyższa wizualizacja prezentuje tylko niewielki zakres możliwości, które oferują sidgety, ale stworzenie jej pozwala na nabycie podstawowych umiejętności, które można wykorzystać do budowy bardziej zaawansowanych stron sieci Web bazujących na tej technologii.

Docelowa wizualizacja pozwalająca na nadzór w czasie rzeczywistym procesu produkcji całej fabryki, w której 4 urządzenia **MultiCon** zbierają dane z czujników zlokalizowanych w różnych jej obszarach, może być podobna do przedstawionej na rysunku 4.7.



Rys. 4.7 Docelowa wizualizacja monitorująca proces produkcyjny

Kod HTML odpowiedzialny za powyższy efekt jest przedstawiony na listingu 4.5.

```

1 <!DOCTYPE html>
2 <html>
3 <head>
4 <title>Factory</title>
5 <script src="http://192.168.1.101/sidgets.js" data-mode=0></script>
6 <style>
7   body {
8     background-color: #cccccc;
9     margin: 0px; }
10  .sidget {
11    --sid-interval: 1;
12    --sid-scaling: 60%; }
13  .dev1 {
14    --sid-color: #00fec3; }
15  .dev2 {
16    --sid-host: '192.168.1.102';
17    --sid-color: #00a2fa; }
18  .dev3 {
19    --sid-host: '192.168.1.103';
20    --sid-color: #fffc09; }
21  .dev4 {
22    --sid-host: '192.168.1.104';
23    --sid-color: #5bcfce; }
24  .chart {
25    --sid-scaling: ''; /*reset scaling*/
26    width: 200px;
27    height: 130px; }
28  #main-container {
29    position: relative;
30    border-width: 0px;
31    margin: auto;
32    width: 1050px; }
33 </style>
34 </head>
35 <body>
36 <div id="main-container">
37   
38   <!-- Device 1 -->
39   <div id="tank1-l" class="sidget dev1" style="--sid-type:12; --sid-chan:1; --sid-scaling:55%; left:7px; top:115px;"></div>
40   <div id="tank1-p" class="sidget dev1" style="--sid-type:9; --sid-chan:4; left:7px; top:285px;"></div>
41   <div id="tank2-l" class="sidget dev1" style="--sid-type:12; --sid-chan:2; --sid-scaling:55%; left:100px; top:62px;"></div>
42   <div id="tank2-p" class="sidget dev1" style="--sid-type:9; --sid-chan:5; left:100px; top:232px;"></div>
43   <div id="tank3-l" class="sidget dev1" style="--sid-type:12; --sid-chan:44; --sid-scaling:55%; left:195px; top:10px;"></div>
44   <div id="tank3-p" class="sidget dev1" style="--sid-type:9; --sid-chan:4; left:195px; top:180px;"></div>
45   <div id="speed" class="sidget dev1" style="--sid-type:8; --sid-chan:37; --sid-color: #ff97be; --sid-scaling:70%; left:282px; top:10px;"></div>
46   <!-- Device 2 -->
47   <div id="flow" class="sidget dev2" style="--sid-type:11; --sid-chan:41; --sid-color: #b3b5ff; --sid-scaling:70%; left:7px; top:550px;"></div>
48   <div id="bootles" class="sidget dev2" style="--sid-type:10; --sid-chan:30; --sid-scaling:70%; left:50px; top:635px;"></div>
49   <!-- Device 3 -->
50   <div id="oxygen" class="sidget dev3 chart" style="--sid-type:3; --sid-chan:35; --sid-color: #ffe05; left:615px; top:10px;"></div>
51   <div id="ph" class="sidget dev3 chart" style="--sid-type:3; --sid-chan:27; --sid-color: #ffe05; left:825px; top:10px;"></div>
52   <div id="temp1" class="sidget dev3" style="--sid-type:4; --sid-chan:20; --sid-scaling:''; width:70px; height:130px; left:740px; top:150px;"></div>
53   <div id="temp2" class="sidget dev3" style="--sid-type:4; --sid-chan:21; --sid-scaling:''; width:70px; height:130px; left:815px; top:165px;"></div>
54   <div id="temp3" class="sidget dev3" style="--sid-type:4; --sid-chan:22; --sid-scaling:''; width:70px; height:130px; left:890px; top:185px;"></div>
55   <div id="temp4" class="sidget dev3" style="--sid-type:4; --sid-chan:23; --sid-scaling:''; width:70px; height:130px; left:965px; top:210px;"></div>
56   <!-- Device 4 -->
57   <div id="temp" class="sidget dev4" style="--sid-type:1; --sid-chan:39; left:745px; top:640px;"></div>
58   <div id="humidit" class="sidget dev4" style="--sid-type:2; --sid-chan:24; left:900px; top:545px;"></div>
59 </div>
60 </body>
61 </html>

```

Listing 4.5. Kod HTML wizualizacji docelowej monitorującej proces produkcyjny

## 4.7. SIDGETY TEKSTOWE

### 4.7.1. Wstęp

Celem nadrzędnym obiektów typu Sidget jest maksymalne uproszczenie procesu tworzenia wizualizacji przez użytkownika. Dlatego też biblioteka graficznych obiektów, spośród których użytkownik może wybierać jest dość bogata.

Istnieją jednak sytuacje, w których może pojawić się potrzeba zoptymalizowania strony z wizualizacją w celu wyświetlenia znacznie większej ilości informacji, wyświetlenia ich w sposób bardziej dopasowany do struktury i kolorystyki docelowego portalu lub nawet zaprojektowania własnych obiektów pokazujących tylko określone dane pobrane z urządzeń. W odpowiedzi na tego typu potrzeby zaprojektowano specjalny typ sidgetu – sidget tekstowy.



Obsługa sidgetu tekstowego została dodana do silnika **MultiCon Sidgets** w firmwarze urządzeń **MultiCon** w wersji **5.14**. Jeśli urządzenie posiada niższą wersję firmware, to chcąc skorzystać z powyższej funkcjonalności, należy najpierw zaktualizować oprogramowanie urządzenia do najnowszej dostępnej wersji.

### 4.7.2. Deklaracja

Sidget tekstowy jest określony jako typ 0 o nazwie Text (patrz parametr `--sid-type` w rozdziale 4.4). Obiekt ten podlega takim samym regułom deklaracji parametrów jak inne typy sidgetów, z nielicznymi wyjątkami dotyczącymi parametrów zorientowanych na wygląd, które w sidgecie tekstowym nie występują. Informacja o braku wsparcia określonego parametru jest umieszczona przy jego opisie.

Podstawową różnicą w stosunku do pozostałych – wstępnie zaprojektowanych sidgetów – sidget tekstowy wymaga określenia w ciele znacznika kontenera dodatkowego kodu HTML użytkownika wraz z wybranymi przez niego zmiennymi. Przykładowa deklaracja znajduje się na listingu 4.6.

```

1 <!DOCTYPE html>
2 <html>
3 <head>
4   <script src="http://192.168.1.100/sidgets.js"></script>
5 </head>
6 <body>
7   <div class="sidget" style="--sid-type:0; --sid-chan:1;">Bieżąca wartość kanału
  "{ChName}" wynosi {Value} {ChUnit} i stanowi {Value;%} zakresu.</div>
8 </body>
9 </html>

```

Listing 4.6 Kod HTML pokazujący sposób deklaracji sidgetu tekstowego

W linii 7 powyższego listingu wewnątrz kontenera `<div>` można zauważyć występowanie specjalnych znaczników: `{ChName}`, `{Value}`, `{ChUnit}`, `{Value;%}`. Są to z góry zdefiniowane zmienne, w miejsce których silnik sidgetów wstawia automatycznie odpowiednie wartości.

Powyższy kod przekłada się na wizualizację, której wygląd jest przedstawiony na rysunku 4.8.

Bieżąca wartość kanału "Przepływ pompy" wynosi 16.8 l/min i stanowi 84% zakresu.

Rys 4.8 Wygląd wizualizacji z wykorzystaniem sidgetu tekstowego

Jak można zauważyć, wyświetlany w przeglądarce tekst jest sformatowany w sposób domyślny. Nic nie stoi na przeszkodzie, by wewnątrz kontenera sidgetu wstawić dowolny kod HTML, który odpowiednio sformatuje zawartość. Przykład większego wykorzystania potencjału sidgetu tekstowego znajduje się na listingu 4.7.

```

1 <!DOCTYPE html>
2 <html>
3 <head>
4   <script src="http://192.168.1.100/sidgets.js"></script>
5   <style>
6     table, th, td { border: 1px solid black; text-align: center;}
7   </style>
8 </head>
9 <body>
10  <table class="sidget" style="--sid-type:0; --sid-chan:1;">
11    <tr> <th>{CN}</th><th>Wartości</th><th>Wartości<br>jako % zakresu {CLL}{CHL} {CU}</th></tr>
12    <tr> <th>{Time-1;HH:mm:ss}</th> <td>{Value-1} {ChUnit}</td> <td>{Value-1;%}</td> </tr>
13    <tr> <th>{Time-2;HH:mm:ss}</th> <td>{Value-2} {ChUnit}</td> <td>{Value-2;%}</td> </tr>
14    <tr> <th>{Time-3;HH:mm:ss}</th> <td>{Value-3} {ChUnit}</td> <td>{Value-3;%}</td> </tr>
15  </table>
16 </body>
17 </html>
18

```

Listing 4.7 Kod HTML pokazujący możliwość formatowania sidgetu tekstowego

Powyższy kod przekłada się na wizualizację, której wygląd jest przedstawiony na rysunku 4.9.

Przepływ pompy	Wartości	Wartości jako % zakresu 0..20 l/min
13:15:46	9.6 l/min	48%
13:15:45	8.1 l/min	40%
13:15:44	6.6 l/min	33%
13:15:43	5.1 l/min	25%

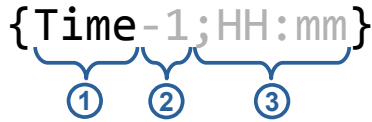
Rys 4.9 Wygląd wizualizacji z wykorzystaniem formatowanego sidgetu tekstowego

W powyższych przykładach deklaracji sidgetu tekstowego wykorzystano wiele zmiennych. Znaczenie każdej z nich jest opisane w podrozdziale 4.7.3.

### 4.7.3. Zmienne

Każda zmienna, którą można zadeklarować w ciele sidgetu tekstowego składa się z nawiasów klamrowych, jej nazwy oraz elementów dodatkowych.

Ogólna struktura zmiennej:



- (1) **Nazwa zmiennej** (element wymagany)
- (2) **Pozycja w buforze** poprzednich wartości (element dodatkowy)
- (3) **Parametr zmiennej** (element dodatkowy)

Nazwa zmiennej może być podana w pełnej lub skróconej formie i jest to element wymagany. Lista dopuszczalnych nazw zmiennych, ich znaczenie i przykłady deklaracji są przedstawione w poniższej tabeli.



Nazwa zmiennej	Nazwa skrócona	Opis zmiennej	Bufor	Parametry	Źródło pozyskania	Przykłady deklaracji wraz z wynikiem
Host	H	Adres IP lub nazwa urządzenia	-	-	--sid-host	{H} → 192.168.3.97 {H} → dev1.multicon.com
Color	C	Kolor wiodący sidgetu	-	-	--sid-color	{Color} → DeepSkyBlue
HiColor	HC	Kolor fragmentu sidgetu określony dla ograniczenia górnego kanału	-	-	--sid-hicolor	{HC} → Red
LoColor	LC	Kolor fragmentu sidgetu określony dla ograniczenia dolnego kanału	-	-	--sid-locolor	{LoColor} → #fff009;
Interval	I	Interwał odświeżania wartości w kanałach podany w sekundach	-	-	--sid-interval	{I} → 300
ChNo	CN	Numer kanału	-	-	--sid-chan	{ChNo} → 1
ChName	CNA	Nazwa kanału	-	-	urządzenie	{CNA} → Przepływ pompy
ChUnit	CU	Jednostka kanału	-	-	urządzenie	{ChUnit} → l/min
ChHiLimit	CHL	Wartość ograniczenia górnego kanału	-	-	urządzenie	{CHL} → 20
ChLoLimit	CLL	Wartość ograniczenia dolnego kanału	-	-	urządzenie	{ChLoLimit} → 0
Value	V	Wartość kanału. Zmienna ta może prezentować ostatnio pobraną lub jedną z wcześniejszych wartości. Może być wyrażona w procentach	TAK	%	urządzenie	{Value} → 15.6 {V;%} → 51% {V-10} → 12.3 {Value-1;%} → 46%
Time	T	Czas wystąpienia wartości w kanale. Zmienna ta może prezentować czas odświeżenia ostatniej lub jednej z poprzednich wartości. Sposób prezentacji czasu można dowolnie dostosować.	TAK	format czasu	web	{Time} → 2021-07-18 13:52:37 {T-60} → 2021-07-18 13:51:37 {T-30;HH:mm:ss} → 13:52:07
BuffPeriod	BP	Informacja o przedziale czasu, dla którego są podawane wartości {ValueMin}, {ValueMax}, {ValueAvg}. Okres ten jest wyliczony na podstawie przyjętego parametru {Interval} i ograniczenia wewnętrznego bufora o wielkości 600 próbek. Deklaracja zmiennej z informacją ograniczającą liczbę próbek pozwala wyliczyć inny przedział czasu.	TAK	-	web	{BP} → 10m {BP-10} → 10s {BuffPeriod-300} → 5m
ValueMin	VMI	Minimalna wartość w kanale, która wystąpiła w okresie {BuffPeriod}. Istnieje możliwość zwrócenia wartości minimalnej, która wystąpiła w krótszym okresie. Można ją również wyrazić w procentach jako wartość przedziału {ChLoLimit}...{ChHiLimit}.	TAK	%	web	{VMI} → 5.0 {VMI;%} → 10% {VMI-10} → 5.3 {VMI-100;%} → 20%
ValueMinTime	VMIT	Czas wystąpienia wartości {ValueMin} w okresie {BuffPeriod}. Istnieje możliwość zwrócenia czasu wystąpienia wartości minimalnej w krótszym okresie. Sposób prezentacji czasu można dowolnie dostosować.	TAK	format czasu	web	{VMIT} → 2021-07-18 13:50:25 {VMIT-60} → 2021-07-18 13:52:03 {VMIT-30;HH:mm:ss} → 13:52:16
ValueMax	VMA	Maksymalna wartość w kanale, która wystąpiła w okresie {BuffPeriod}. Istnieje możliwość zwrócenia wartości maksymalnej, która wystąpiła w krótszym okresie. Można ją również wyrazić w procentach jako wartość przedziału {ChLoLimit}...{ChHiLimit}.	TAK	%	web	{ValueMax} → 20.0 {VMA;%} → 110% {VMA-10} → 16.2 {VMA-100;%} → 95%
ValueMaxTime	VMAT	Czas wystąpienia wartości {ValueMax} w okresie {BuffPeriod}. Istnieje możliwość zwrócenia czasu wystąpienia wartości maksymalnej w krótszym okresie. Sposób prezentacji czasu można dowolnie dostosować.	TAK	format czasu	web	{VMAT} → 2021-07-18 13:50:27 {VMAT-60} → 2021-07-18 13:52:06 {VMAT-30;HH:mm:ss} → 13:52:27
ValueAvg	VA	Wartość średnia z pomiarów, które wystąpiły w okresie {BuffPeriod}. Istnieje możliwość zwrócenia wartości średniej wyliczonej na podstawie krótszego okresu. Można ją również wyrazić w procentach jako wartość przedziału {ChLoLimit}...{ChHiLimit}.	TAK	%	web	{ValueAvg} → 15.5 {VA;%} → 48% {VA-10} → 13.3 {ValueAvg-300;%} → 51%

W przypadku zmiennych prezentujących czas, jako parametr możliwe jest podanie ciągu specyfikatorów formatowania czasu. Lista tych specyfikatorów wraz z ich znaczeniem jest przedstawiona w poniższej tabeli.

Specyfikator formatowania	Opis
<b>D</b>	Dzień miesiąca od 1 do 31
<b>DD</b>	Dzień miesiąca od 01 do 31
<b>DDD</b>	Skrócona nazwa dnia tygodnia
<b>DDDD</b>	Pełna nazwa dnia tygodnia
<b>M</b>	Miesiąc, od 1 do 12
<b>MM</b>	Miesiąc, od 01 do 12
<b>MMM</b>	Skrócona nazwa miesiąca
<b>MMMM</b>	Pełna nazwa miesiąca
<b>Y</b>	Rok, od 0 do 99
<b>YY</b>	Rok, od 00 do 99
<b>YYY</b>	Rok z co najmniej trzema cyframi
<b>YYYY</b>	Rok jako liczba czterocyfrowa
<b>h</b>	Godzina, przy użyciu zegara 12-godzinnego od 1 do 12
<b>hh</b>	Godzina, przy użyciu zegara 12-godzinnego od 01 do 12
<b>H</b>	Godzina, przy użyciu zegara 24-godzinnego od 0 do 23
<b>HH</b>	Godzina, przy użyciu zegara 24-godzinnego od 00 do 23
<b>m</b>	Minuta, od 0 do 59
<b>mm</b>	Minuta, od 00 do 59
<b>s</b>	Sekunda, od 0 do 59
<b>ss</b>	Sekunda, od 00 do 59
<b>t</b>	Pierwszy znak oznaczenia am/pm (małe litery)
<b>tt</b>	Oznaczenie am/pm (małe litery)
<b>T</b>	Pierwszy znak oznaczenia AM/PM (duże litery)
<b>TT</b>	Oznaczenie AM/PM (duże litery)
<b>K</b>	Informacje o strefie czasowej
<b>z</b>	Przesunięcie godzin względem UTC, bez zer wiodących
<b>zz</b>	Przesunięcie godzin względem UTC, z wiodącym zerem dla wartości jednocyfrowej
<b>zzz</b>	Godziny i minuty przesunięte względem UTC

## 5. KOMPATYBILNOŚĆ

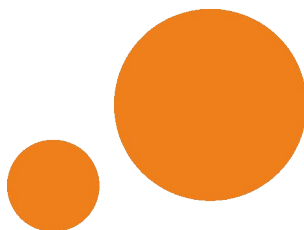
Silnik **MultiCon Sidgets** wykorzystuje technologie HTML 5, CSS 3 i JavaScript ES6. O poprawne działanie tych technologii dbają odpowiednio opracowane silniki przeglądarek internetowych. Ze względu na zapewnienie właściwego działania zaprojektowanych wizualizacji oraz bezpieczeństwo systemów operacyjnych, zaleca się korzystać z przeglądarek internetowych zaktualizowanych do najnowszych wersji.



*Poprawność działania silnika sidgetów została sprawdzona przez producenta na wybranych przeglądarkach w określonych wersjach. Producent gwarantuje jego właściwe działanie tylko na sprawdzonych przez niego wersjach przeglądarek. Niemniej jednak istnieje małe ryzyko, że na nowszych wersjach tych programów silnik **MultiCon Sidgets** nie będzie działał lub będzie działał nieprawidłowo.*

**Sprawdzone wersje przeglądarek internetowych:**

Nazwa przeglądarki	Wersja przeglądarki	System operacyjny	Działa?
<b>DAQ Manager</b>	do 1.9	Windows XP-10	<b>NIE</b>
<b>DAQ Manager</b>	od 1.10	Windows XP-10	<b>TAK</b> (zewnętrzna przeglądarka)
<b>Microsoft Internet Explorer</b>	11.0	Windows XP-10	<b>NIE</b>
<b>Microsoft Edge</b>	44.18362	Windows 10	<b>TAK</b>
<b>Google Chrome</b>	79.0	Windows	<b>TAK</b>
<b>Mozilla Firefox</b>	71.0	Windows	<b>TAK</b>
<b>Google Chrome</b>	78.0	Android 9	<b>TAK</b>
<b>Google Chrome</b>	87.0	Android 9	<b>TAK</b>
<b>Google Chrome</b>	91.0	Android 9 i 10	<b>TAK</b>
<b>Mozilla Firefox</b>	87.0	Android	<b>TAK</b>
<b>Mozilla Firefox</b>	88.1	Android 9	<b>TAK</b>
<b>Apple Safari</b>	12.1	iOS 12.4.1	<b>TAK</b>



**SIMEX Sp. z o.o.  
ul. Wielopole 11  
80-556 Gdańsk  
Poland**

**tel.: (+48 58) 762-07-77  
fax: (+48 58) 762-07-70**

**<http://www.simex.pl>  
e-mail: [info@simex.pl](mailto:info@simex.pl)**