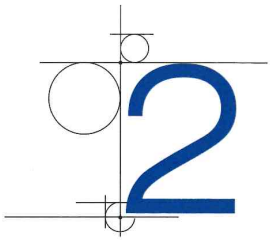


# Framework Angular

**Angular** to obok React, Vue.js i Express jeden z najpopularniejszych frameworków JavaScript, których użycie pozwala stworzyć aplikację webową. Należy on do grupy tzw. frameworków SPA (ang. *Single Page Application*). Główne okno aplikacji ładowane jest tylko raz, a odświeżane są jedynie wymagane fragmenty, i nie jest już konieczne przeładowanie całej strony, gdy następuje zmiana widoku, np. podczas przejścia pomiędzy podstronami.

## 2.1. Angular — instalacja i konfiguracja środowiska pracy

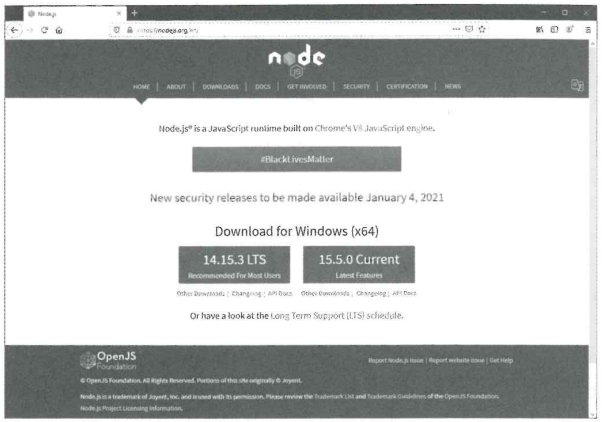
### 2.1.1. Przygotowanie środowiska

#### Instalacja Node.js

Konfigurację środowiska pracy rozpoczynamy od instalacji **Node.js**. Node.js jest otwartoźródłowym oprogramowaniem, które pozwala uruchomić kod JavaScript nie lokalnie, w oknie przeglądarki (jak dzieje się zazwyczaj), lecz po stronie serwera. Środowisko to jest szerzej opisane w dalszej części podręcznika.

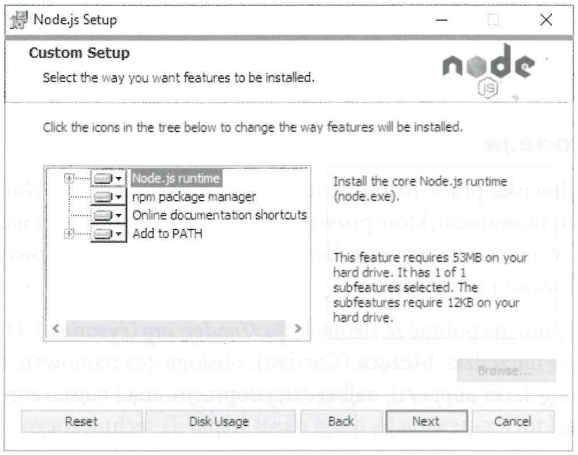
Pakiet instalacyjny można pobrać ze strony *https://nodejs.org* (rysunek 2.1). Udostępnione są zawsze dwie wersje narzędzia: **bieżąca** (*Current*), obsługująca najnowsze funkcje i dodatki, oraz **LTS** (ang. *Long Term Support*), najbardziej dopracowana i najstaranniej przetestowana przez producenta, który zapewnia jej długi okres wsparcia technicznego.

Ponieważ framework Angular jest bardzo często uaktualniany, przed samą instalacją warto się zapoznać z jego wymaganiami, dostępnymi na stronie *https://angular.io/guide/setup-local*.




**Rysunek 2.1.** Okno strony internetowej, z której należy pobrać instalator Node.js

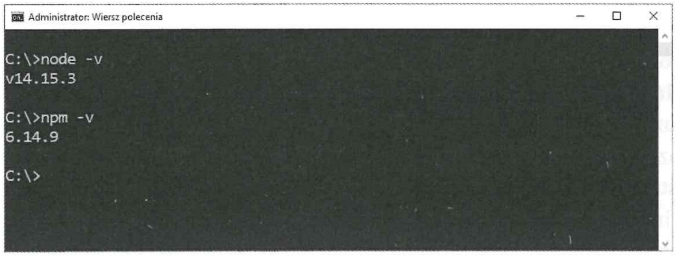
Instalacja Node.js nie różni się od sposobu, w jaki instaluje się inne programy (rysunek 2.2). Po zaakceptowaniu licencji, określeniu ścieżki docelowej i pozostawieniu opcji domyślnie zaproponowanych przez instalator nastąpi zainstalowanie oprogramowania w systemie. Wraz z Node.js zostanie zainstalowany **menedżer pakietów npm**, który posłuży do zainstalowania i pobrania niezbędnych funkcjonalności wykorzystywanych przez projekt.




**Rysunek 2.2.** Okno instalacji Node.js — wybór komponentów

Weryfikację instalacji można przeprowadzić za pomocą konsoli systemu Windows. W tym celu należy użyć poleceń `node -v` oraz `npm -v`. Pierwsze sprawdza wersję Node.js, a drugie — wersję menedżera pakietów npm (rysunek 2.3).



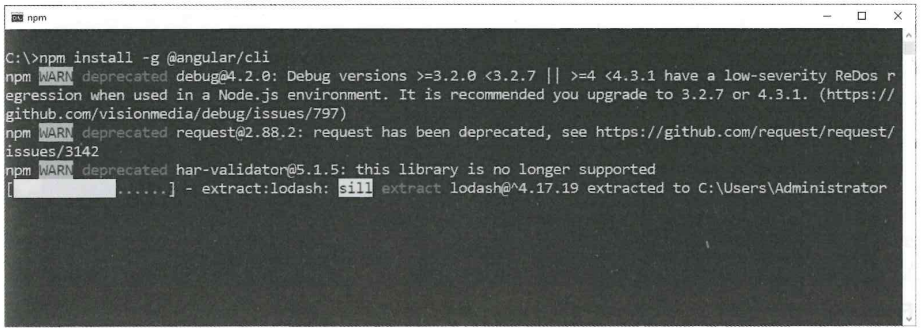



**Rysunek 2.3.** Weryfikacja procesu instalacji Node.js i npm

## Instalacja Angular CLI

Gdy wszystko przebiegło pomyślnie, możemy przejść do następnego kroku, czyli instalacji **Angular CLI** — narzędzia, które pozwoli utworzyć „szkielet" aplikacji. We wcześniejszych wersjach Angulara niezbędne komponenty i moduły odpowiedzialne za zbudowanie aplikacji należało przygotować samemu, jednak Angular CLI wszystko to automatyzuje, dzięki czemu za pomocą jednego polecenia utworzymy i skonfigurujemy nowy projekt.

Aby zainstalować Angular CLI, należy wydać polecenie `npm install -g @angular/cli` (rysunek 2.4).




**Rysunek 2.4.** Instalacja narzędzia Angular CLI

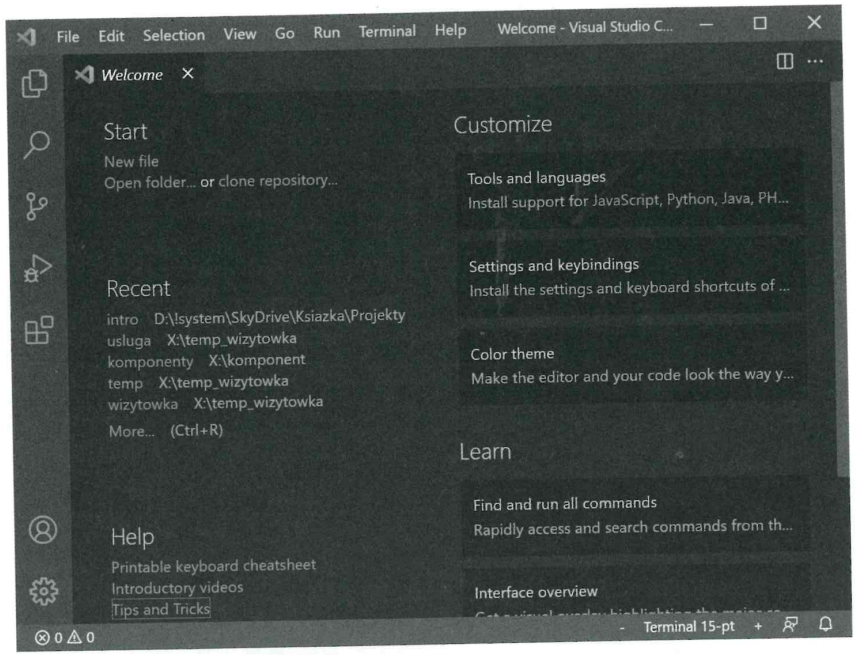
**UWAGA**

Ponieważ działanie Angulara opiera się na innych modułach, podczas instalacji mogą zostać wyświetlone ostrzeżenia, których źródłem są użyte moduły. Informują one najczęściej o zmianie sposobu działania lub zalecają aktualizację do nowszej wersji.

Użycie w poleceniu flagi `-g` spowoduje, że Angular CLI zostanie **zainstalowany globalnie**. Dzięki dodaniu odpowiedniego wpisu w zmiennej *PATH* systemu polecenia odnoszące się do narzędzia będzie można wydawać niezależnie od bieżącego położenia w strukturze katalogów.

## Instalacja Visual Studio Code

Ostatnim krokiem jest instalacja edytora kodu. Jednym z najwyżej ocenianych jest darmowy **Visual Studio Code** (rysunek 2.5). Aplikacja jest niezwykle popularna dzięki dużej społeczności skupionej wokół niej, co zaowocowało powstaniem ogromnej liczby dodatkowych wtyczek i rozszerzeń ułatwiających pracę z kodem. Kolejnym atutem narzędzia jest to, że zostało udostępnione w wersjach przeznaczonych na wszystkie najważniejsze platformy systemowe (Windows, Linux oraz macOS). Dlatego kod aplikacji internetowych będziemy pisać z użyciem właśnie tego edytora. Można go pobrać ze strony *https://code.visualstudio.com/*.




**Rysunek 2.5.** Okno programu Visual Studio Code

**CIEKAWOSTKA**

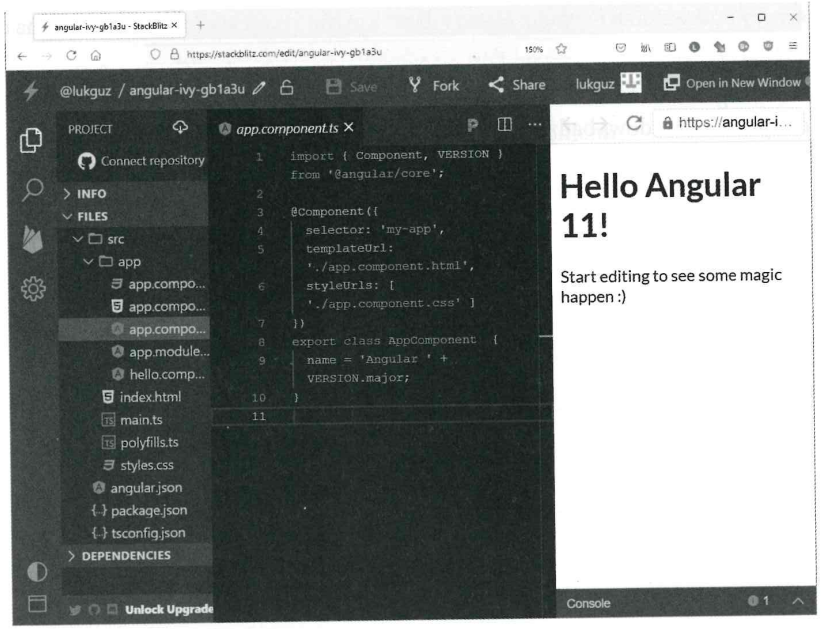
Za wyborem Visual Studio Code przemawia jeszcze jeden atut — producentem narzędzia jest firma Microsoft, która jest także zaangażowana w rozwój języka programowania TypeScript. Tego języka użyjemy wraz z frameworkiem Angular do napisania własnej aplikacji webowej. Oznacza to również, że jeżeli zachodzi jakakolwiek zmiana w języku TypeScript, jest ona bardzo szybko odzwierciedlana w samym edytorze.

Visual Studio Code nie jest oczywiście jedynym dostępnym edytorem kodu. Spośród darmowych do najczęściej wykorzystywanych należą **Atom** (*https://atom.io/*) i **Brackets** (*http://brackets.io/*).



## StackBlitz — zdalne środowisko pracy

Pokazane rozwiązanie zostało oparte na instalacji lokalnej, ale są też takie, które umożliwiają pracę z kodem z dowolnego miejsca i w każdym systemie operacyjnym. Na pracę zdalną i w szerszej grupie osób pozwala platforma **StackBlitz**, dostępna pod adresem: *https://stackblitz.com/* (rysunek 2.6).




**Rysunek 2.6.** Platforma StackBlitz

StackBlitz nie tylko pozwala kodować z użyciem frameworka Angular, ale także oferuje przestrzenie robocze (tzw. *workspace*) dla wielu innych języków i frameworków, np. HTML/CSS/JavaScript, React czy Node.js.

Przedstawione rozwiązania (lokalne i online) łączy to, że w obu użyto edytora Visual Studio Code — w pierwszym — jako programu działającego pod kontrolą systemu, w drugim zaś — jako aplikacji wyświetlanej w oknie przeglądarki. Z obu rozwiązań można korzystać równolegle.

**WSKAZÓWKA**

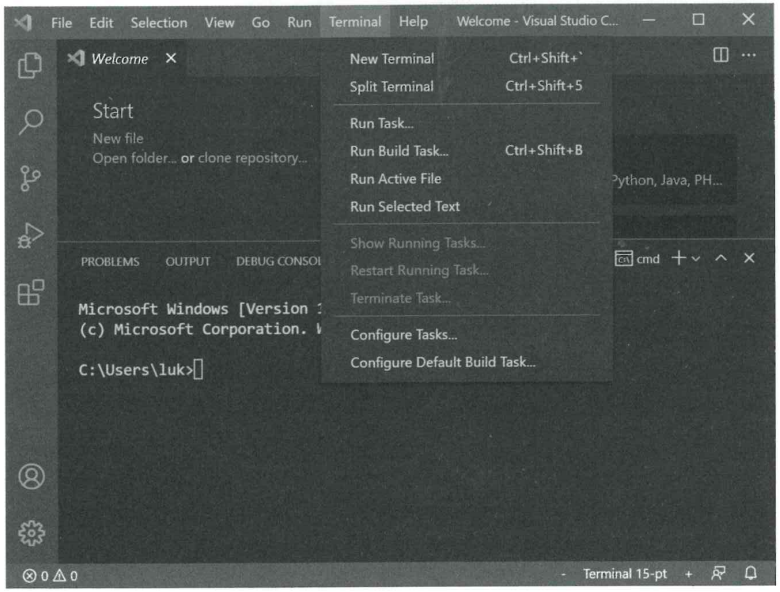
Po zarejestrowaniu się i powiązaniu konta z serwisem GitHub będzie można kodować online, a napisany kod udostępnić innym osobom, np. w sytuacji, gdy trzeba znaleźć rozwiązanie problemu.

## 2.1.2. Generowanie projektu Angular

Generowanie pierwszego projektu należy rozpocząć od utworzenia katalogu, w którym będą przechowywane pliki. Nazwa katalogu może być dowolna, ale tak jak w przypadku innych języków programowania, wszędzie, gdzie jest to możliwe, **unikamy stosowania polskich znaków oraz spacji** (użyte wyrazy łączymy ze sobą znakiem łącznika bądź podkreślenia).

Na potrzeby podręcznika przyjęto, że wszystkie tworzone projekty zostały zapisane na dysku *D* w katalogu *projekty*.

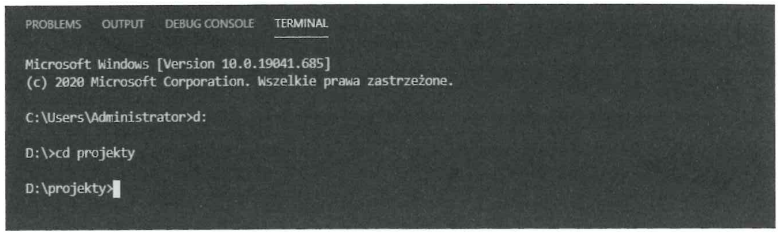
Ponieważ Angular CLI to narzędzie wiersza poleceń, obsługuje się je za pomocą wiersza poleceń systemu Windows bądź konsoli Windows PowerShell. Najczęściej jednak stosowane jest najwygodniejsze rozwiązanie — terminal zintegrowany z edytorem kodu. Tę funkcjonalność zapewnia również Visual Studio Code. Aby w oknie edytora wyświetlić widok terminala, z górnego menu wybieramy *Terminal*, a następnie *New Terminal* (rysunek 2.7) bądź korzystamy ze skrótu klawiszowego *Ctrl+`* (użycie skrótu *Ctrl+Shift+`* spowoduje dodanie kolejnego okna terminala).




**Rysunek 2.7.** Visual Studio Code — włączenie okna terminala

W oknie uruchomionego terminala poleceniem `cd` przechodzimy do katalogu, w którym utworzony zostanie projekt (rysunek 2.8).






**Rysunek 2.8.** Użycie polecenia cd — przejście do katalogu, w którym zostanie utworzony projekt Angular

Aby wygenerować nowy projekt, należy posłużyć się poleceniem:

```
ng new <opcje_dodatkowe> <nazwa_projektu>
```

gdzie:

- `opcje_dodatkowe` — przełączniki pozwalające włączyć/wyłączyć funkcje tworzonego projektu. Do takich przełączników należy np. `--minimal`. Jego dodanie spowoduje, że projekt będzie zawierał tylko pliki niezbędne do utworzenia aplikacji. Pełna lista opcji zostanie wyświetlona po użyciu przełącznika `--help`.
- `nazwa_projektu` — nazwa tworzonego projektu, bez polskich znaków.

Po wydaniu polecenia zostanie uruchomiony kreator, który zada nam kilka pytań.

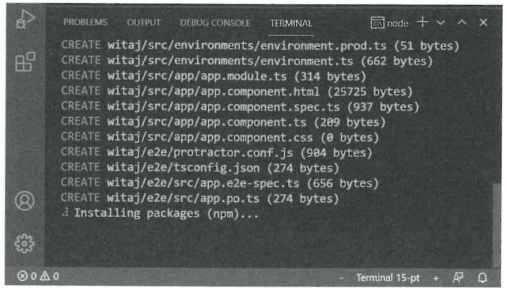
Pierwsze pytanie brzmi: `Do you want to enforce stricter type checking and stricter bundle budgets in the workspace? This setting helps improve maintainability and catch bugs ahead of time. For more information, see https://angular.io/strict (y/N).`

Odpowiedź twierdząca uruchomi tzw. **tryb ścisły** (ang. *strict mode*). Zostanie włączony mechanizm kontroli kodu, który bardziej restrykcyjnie weryfikuje tworzony kod.

Drugie pytanie to: `Would you like to add Angular routing? (y/N)`. Wpisanie `y` spowoduje uaktywnienie funkcji **routingu**. Na razie ta funkcjonalność nie jest wymagana, dlatego można odpowiedzieć `N`.

Następnie zostanie wyświetlona lista wyboru — należy wskazać preprocesor CSS (język skryptowy, który jest interpretowany lub kompilowany do kaskadowych arkuszy stylów). Wybieramy opcję domyślną — `CSS` (Angular pozwala również formatować treści z użyciem `SCSS`, `Sass`, `Less` lub `Stylus`).

Po odpowiedzeniu na pytania rozpocznie się proces tworzenia projektu. Jego postęp możemy obserwować w oknie terminala. Zainstalowany wraz z Node.js menedżer plików npm pobierze wszystkie niezbędne komponenty (rysunek 2.9).




**Rysunek 2.9.** Tworzenie projektu Angular

Pomyślne utworzenie projektu powinno się zakończyć wyświetleniem komunikatu `Packages installed successfully`.

**UWAGA**

Ponieważ Angular CLI jest nieustannie rozwijany, wraz z udostępnianiem nowych wersji narzędzia lista pytań może się zmieniać.

Ostatnią czynnością jest sprawdzenie, czy wygenerowany projekt zostanie poprawnie uruchomiony. W tym celu za pomocą edytora otwieramy zawartość katalogu, którego nazwa jest tożsama z nazwą projektu. Aby otworzyć katalog, należy z górnego menu wybrać opcję *File*, a następnie *Open Folder* i w nowo otwartym oknie określić jego położenie. Drugi sposób polega na przeciągnięciu katalogu zawierającego projekt i upuszczeniu go w oknie Visual Studio Code. Lista plików składająca się na projekt zostanie wyświetlona z lewej strony okna edytora.

Uruchomienie projektu następuje po otwarciu terminala i wydaniu polecenia `ng serve`.

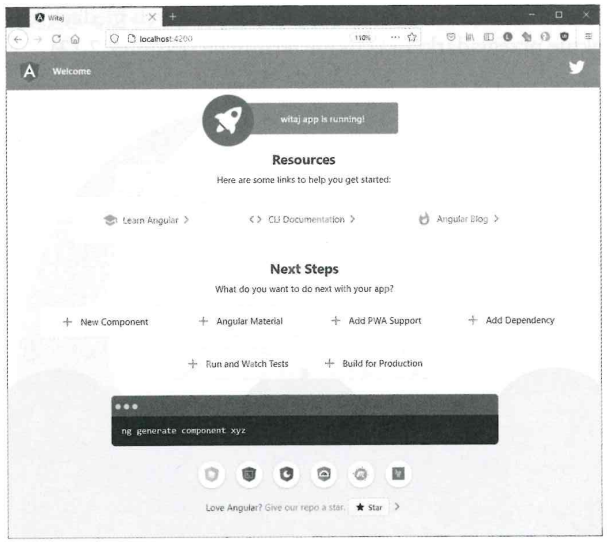
Po udanym skompilowaniu projektu zostanie uruchomiony serwer hostujący aplikację. Aby sprawdzić jej działanie, w oknie przeglądarki należy wpisać adres `localhost:4200`, gdzie port 4200 to domyślny port, pod którym działa aplikacja Angular (rysunek 2.10). Alternatywną opcją jest zastosowanie w poleceniu `ng serve` flagi `–o`. Jej dodanie spowoduje automatyczne otwarcie okna domyślnej przeglądarki.

**WSKAZÓWKA**

Aby zmienić domyślny port usługi, np. na 5000, należy do polecenia uruchamiającego aplikację dodać: `--port 5000`. Alternatywną opcją jest dodanie w pliku *angular.json* do obiektu `options` znajdującego się w `projects – <nazwa_projektu> – architect – serve` dwóch kluczy z wartościami: `"host": "127.0.0.1"` oraz `"port": 5000` (oddzielonych od siebie przecinkami).

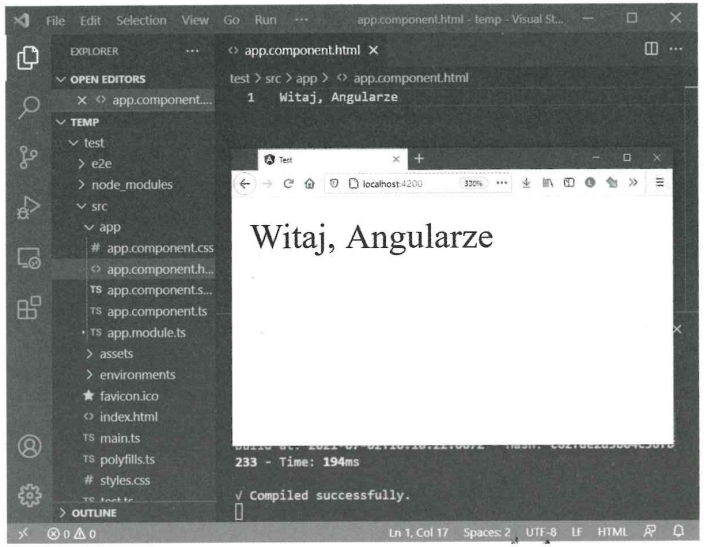
Aplikacja została poprawnie wygenerowana i uruchomiona



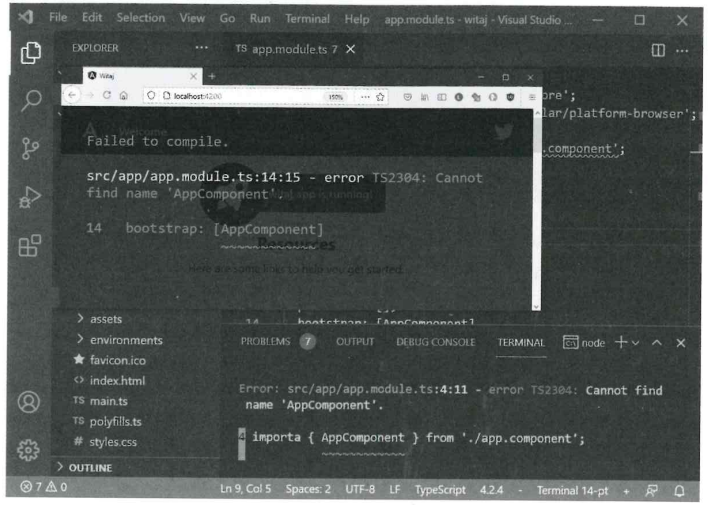



**Rysunek 2.10.** Domyślny widok okna aplikacji Angular

Uruchomiony serwer działa w **trybie deweloperskim**, co oznacza, że po zapisaniu zmian wprowadzonych w kodzie stan aplikacji jest na bieżąco aktualizowany w oknie przeglądarki. Aby przetestować działanie tego trybu, w drzewie plików odszukujemy katalog *app*, a w nim plik *app.component.html*. Po wyświetleniu zawartości pliku zastępujemy ją np. ciągiem: *Witaj, Angularze*. Zapisanie pliku wymusi zmianę w oknie przeglądarki (rysunek 2.11).



W trakcie tworzenia kodu aplikacji komunikat o napotkanych błędach kompilacji zostanie wyświetlony w oknie przeglądarki oraz oknie terminala (rysunek 2.12).




**Rysunek 2.12.** Błędy kompilacji aplikacji Angular — terminal, okno przeglądarki

**Pytania kontrolne**

1. Jakie narzędzia są niezbędne, aby można było utworzyć projekt Angular?
2. Jakie korzyści daje instalacja globalna? Czy taki sposób instalacji ma wady?
3. Co oznacza, że serwer działa w trybie deweloperskim?
4. Porównaj ze sobą lokalne i zdalne środowiska pracy. Wymień ich wady i zalety.

# 2.2. TypeScript

**TypeScript** to język programowania będący nadzbiorem języka JavaScript. To oznacza, że obowiązuje w nim składnia języka JavaScript, ale rozbudowana o dodatkowe elementy. Dzięki temu kod napisany w TypeScripcie jest czytelniejszy, a liczba popełnianych błędów — mniejsza. Wykorzystują go m.in.: Facebook, Microsoft, Google i Netflix. Mieści się on w pierwszej dziesiątce najchętniej wybieranych języków programowania, a jego popularność ciągle wzrasta (według PYPL PopularitY of Programming Language — *https://pypl.github.io/PYPL.html*).



## 2.2.1. Pierwsze użycie TypeScript. Instalacja niezbędnych narzędzi

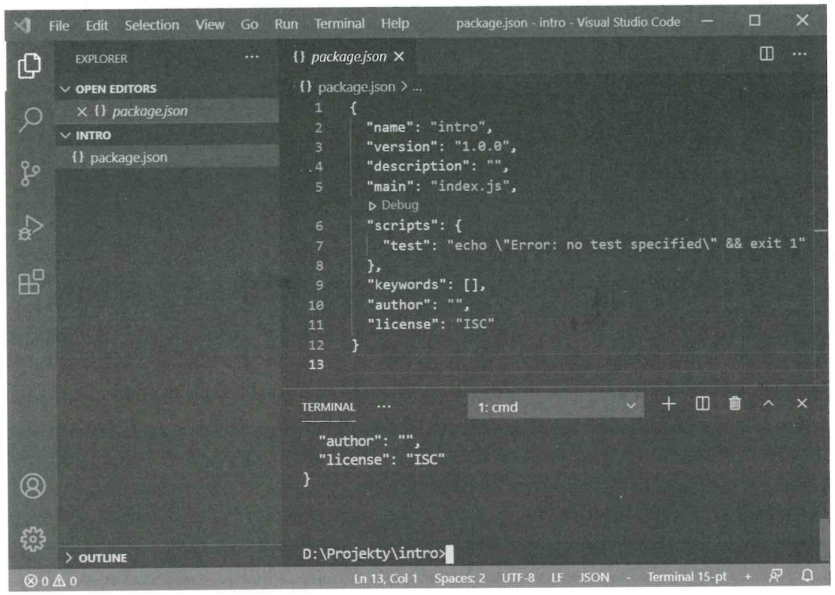
### Przygotowanie środowiska pracy — TypeScript

TypeScript jest wykorzystywany w połączeniu z Angularem, Node.js czy jako zamiennik czystego JavaScriptu. Kod zapisany w TypeScripcie, aby mógł zostać wykonany przez przeglądarkę, trzeba przekonwertować na JavaScript.

Na samym początku nauki tego języka można by wykorzystać projekt utworzony w poprzednim rozdziale, ale ponieważ użycie TypeScriptu nie ogranicza się tylko do Angulara, często trzeba użyć środowiska, które pozwoli pracować z „czystym" językiem.

Budowanie projektu rozpoczynamy od utworzenia katalogu, w którym będziemy przechowywać jego pliki. W tym celu w lokalizacji *D:\Projekty* został utworzony katalog o nazwie *intro*.

W terminalu po uruchomieniu Visual Studio Code należy przejść do katalogu projektu i wydać polecenie `npm init -y` (po uprzedniej instalacji Node.js). Po zatwierdzeniu polecenia zostanie utworzony plik *package.json*, w którym będą zapisane podstawowe informacje o tworzonym projekcie, m.in. jego nazwa, wersja, opis (rysunek 2.13). Pominięcie flagi `-y` uruchomi kreator, który zapyta nas o te dane.




**Rysunek 2.13.** Podstawowe informacje o projekcie — plik package.json