PROGRAM SZKOLENIA NARZĘDZIA I DEVOPS

Szkolenie Biblioteki npm i CLI

Wydasz własną paczkę npm albo narzędzie CLI, które instaluje się bez niespodzianek i łatwo je utrzymać.

CZAS TRWANIA	POZIOM	FORMA
2 dni	**średniozaawansowany**	**warsztat dla zespołu, zdalnie lub stacjonarnie**

Wydzielenie kodu do osobnej paczki wygląda na prostą rzecz: `npm init`, `npm publish` i gotowe. Potem okazuje się, że w paczce nie ma skompilowanego kodu, typy nie działają w projekcie z ESM, narzędzie CLI nie uruchamia się u kolegi na Windowsie, publikacja z CI przestała działać po zmianach w tokenach npm, a nikt nie pamięta, co zmieniło się między wersjami.

Na szkoleniu zbudujesz od zera bibliotekę w TypeScript i narzędzie CLI, a potem wydasz je tak, jak wydają je dojrzałe projekty open source: z testami, changelogiem, publikacją z CI bez tokenu w sekretach i listą kontrolną, którą sprawdzasz przed każdym wydaniem. Szablon projektu dostajesz przed szkoleniem, więc od pierwszej godziny piszesz kod paczki.

PROGRAM

2 dni

- Anatomia paczki
  - `package.json`: `name`, `version`, `exports`, `main`, `types`
  - Pola `bin`, `engines` i `private`
  - `dependencies`, `peerDependencies` i kiedy które
  - Kiedy wydać wersję major: semver z perspektywy autora paczki
  - ESM-only czy dual package: `require(esm)` bez flagi od Node.js 22.12 i 20.19, warunek `module-sync` w `exports`
- Projekt biblioteki z gotowego szablonu
  - TypeScript, build do `dist/`, `.nvmrc`
  - Vitest, ESLint i Prettier
  - Type stripping w Node.js nie działa w `node_modules`, więc wydajesz JS
- Co naprawdę trafia do paczki i jak to przetestować
  - Pole `files`, `npm pack --dry-run` i podgląd archiwum
  - Skrypt `prepack`, czyli paczka bez skompilowanego kodu nigdy więcej
  - Sprawdzanie typów i eksportów: `publint`, `@arethetypeswrong/cli`
  - `npm link` i instalacja z lokalnego archiwum

- Narzędzie CLI
  - Katalog `bin/` z cienkim plikiem startowym, logika w `src/`
  - Argumenty przez `parseArgs` z `node:util`, pomoc i komunikaty błędów
  - Kody wyjścia, `stdout` kontra `stderr`, kolory i `NO_COLOR`
  - Konfiguracja: flagi, zmienne środowiskowe i plik `.env`
  - Cache i konfiguracja w katalogach systemu, np. XDG na Linuksie
  - Binarka działająca na Windows, macOS i Linux, testy uruchamiające ją
- Wydanie wersji
  - `npm version`, tagi w Gicie i automatyczny `CHANGELOG.md`
  - Tagi dystrybucyjne (`next`, `beta`), `npm deprecate`
  - README, które sprzedaje paczkę: instalacja, przykład, demo w formie GIF-a
- Uwierzytelnianie i publikacja z CI
  - Koniec tokenów klasycznych w grudniu 2025, tokeny granularne z prawem zapisu ważne do 90 dni, 2FA i dwugodzinne sesje `npm login`
  - Trusted publishing z GitHub Actions (npm 11.5.1+, Node.js 22.14+), także dla prywatnych paczek; automatyczne provenance tylko dla publicznej paczki z publicznego repozytorium
  - Staged publishing (npm 11.15+): CI wysyła wersję do kolejki, a Ty zatwierdzasz ją z 2FA; od 2026-09-03 nowy trusted publisher domyślnie pozwala tylko na staging, bezpośredni `npm publish` włączasz osobno
  - Pierwsza wersja nowej paczki wydana ręcznie, bo trusted publishera ustawiasz dla paczki, która już istnieje w rejestrze
  - Prywatny rejestr (GitHub Packages, Verdaccio, Artifactory): publikacja tokenem, bo trusted publishing działa w npmjs.com
  - Matryca wspieranych wersji LTS Node.js i zgodne pole `engines`
- Utrzymanie i bezpieczeństwo po wydaniu
  - Łańcuch dostaw: przejęte konta i złośliwe wersje paczek
  - `npm audit signatures`, weryfikacja provenance, skrypty instalacyjne
  - Aktualizacje zależności paczki: Dependabot i Renovate
  - Licencja i lista kontrolna przed publikacją

Więcej o konfiguracji CI znajdziesz na szkoleniu CI/CD z GitHub Actions.

DLA KOGO?

- Dla programistów, którzy chcą wydzielić wspólny kod do paczki używanej w kilku projektach firmy.
- Dla zespołów, które piszą narzędzia wewnętrzne i skrypty uruchamiane z terminala.
- Dla osób, które chcą wydać swoją pierwszą bibliotekę open source.

CO MUSISZ UMIEĆ?

- Swobodnie pisać w JavaScript (podstawy TypeScript są mile widziane).
- Uruchamiać projekty w Node.js i korzystać z npm.

- Pracować z Gitem na co dzień.
- Mieć konto na npmjs.com z włączonym 2FA i konto na GitHubie (publikację ćwiczysz na własnej paczce testowej).

CZEGO SIĘ NAUCZYSZ?

- Zbudujesz bibliotekę w TypeScript, która działa zarówno z `import`, jak i z `require`.
- Dobierzesz `dependencies` i `peerDependencies` tak, żeby paczka nie dublowała bibliotek w projekcie użytkownika.
- Napiszesz narzędzie CLI z czytelną pomocą, poprawnymi kodami wyjścia i cache, działające na każdym systemie.
- Sprawdzisz przed publikacją, co dokładnie trafi do paczki.
- Wydasz nową wersję z changelogiem jedną komendą.
- Wydasz ręcznie pierwszą wersję paczki, a kolejne skonfigurujesz do publikacji z GitHub Actions przez trusted publishing, bez tokenu npm w sekretach: bezpośrednio albo przez staging z zatwierdzeniem z 2FA. Dla rejestru spoza npmjs.com dobierzesz token.
- Ustawisz automatyczne aktualizacje zależności paczki.
- Ochronisz paczkę i jej użytkowników przed atakami na łańcuch dostaw.

OPINIE UCZESTNIKÓW

> *„Gruntownie, krok po kroku stworzyliśmy aplikację sklepu internetowego, po drodze pisząc testy jednostkowe, e2e, web komponenty, autorski backend oparty na Node.js + Express.js, jak również wykorzystaliśmy narzędzia CI.”*
>
> Łukasz, JavaScript Developer (WarsawJS MasterClass)

> *„Piotr przekazywał wiedzę na temat tworzenia odpornych na błędy aplikacji internetowych. Podczas szkolenia tworzyliśmy testy jednostkowe, testy end-to-end aplikacji po stronie klienta oraz testy backendu napisanego w Node.js.”*
>
> Dominik, Frontend Developer (WarsawJS MasterClass)

TRENER

Piotr Kowalski

W branży IT od 2008 roku. Od 2014 jest mentorem dla wielu developerów. Jako trener spędził na sali szkoleniowej ponad 1000 godzin, a na LinkedIn ma 52 rekomendacje. Każde jego szkolenie to dawka solidnej wiedzy oraz humoru.

Więcej: piecioshka.pl/piotr-kowalski

ZORGANIZUJMY TO SZKOLENIE W TWOIM ZESPOLE

Program dopasuję do Waszego projektu i poziomu grupy. Napisz, z czym pracujecie, a odpowiem z propozycją terminu i zakresu.

KONTAKT	**piecioshka.pl/kontakt**
E-MAIL	**piecioshka+blog [małpka] gmail.com**
SZKOLENIE	**piecioshka.pl/szkolenia/biblioteki-npm/**