5 Podstawy współbieżności

W tym rozdziale zostaną omówione następujące tematy:

- Rozumienie zasad współbieżności BEAM.
- Praca z procesami.
- Praca z procesami serwerów stanowych.
- Dyskusja na temat czasu uruchomienia.

Kiedy już masz odpowiedni zasób wiedzy na temat Elixira i funkcyjnych zwrotów programowania, przejdziemy do platformy Erlanga. Poświęcimy trochę czasu na zapoznanie się ze współbieżnością BEAM, funkcją pełniącą kluczową rolę dla Elixira i Erlanga w kontekście skalowalności, odporności na błędy i rozproszenia.

W tym rozdziale rozpoczniemy przygodę ze współbieżnością BEAM, poznając podstawowe metody i narzędzia. Zanim przejdziemy do szczegółów, przyjrzyjmy się kluczowym zasadom.

5.1. Współbieżność w BEAM

Założeniem związanym z Erlangiem jest tworzenie wysoce dyspozycyjnych systemów, które mogą działać bez przerwy i są w stanie zawsze właściwie odpowiedzieć na żądania klienta. Aby stworzyć wysoce dyspozycyjny system, musisz poradzić sobie z następującymi trudnościami:

- *Odporność na błędy* – Przywracaj do właściwego stanu, ograniczaj i izoluj skutki błędów czasu uruchomienia.
- *Skalowalność* – Radź sobie ze zwiększonym obciążeniem poprzez dodawanie zasobów sprzętowych bez reorganizowania kodu.
- *Rozproszenie* – Uruchamiaj swój system na wielu urządzeniach, tak aby inne mogły przejąć obowiązki tego, które przestało funkcjonować.

Jeżeli zaradzisz powyższym problemom, twój system będzie zapewniać usługi stale, praktycznie bez przestojów i błędów. Współbieżność odgrywa istotną rolę w procesie osiągania wysokiej dyspozycyjności. W BEAM jednostką współbieżności jest *proces*: podstawowy budulec umożliwiający tworzenie skalowalnych, odpornych na błędy systemów rozproszonych.

UWAGA Proces BEAM to nie to samo, co proces systemu operacyjnego. Procesy BEAM są znacznie lżejsze i tańsze niż procesy systemu operacyjnego, o czym przekonasz się za chwilę. Książka ta traktuje głównie o BEAM, toteż pojęcie *procesu* w dalszej części tekstu będzie dotyczyć jedynie procesów BEAM.

Na etapie produkcji typowy system serwera powinien być w stanie obsłużyć jednoczesne żądania wielu różnych klientów, zapewnić stan współdzielony (np. pamięć podręczna, dane sesji użytkownika oraz dane z całego serwera) i wykonywać dodatkowe zadania w tle. Serwer będzie działać poprawnie, jeżeli wszystkie te zadania zostaną wykonywane szybko i skutecznie.

Ze względu na to, że wiele zadań jednocześnie oczekuje na wykonanie, bardzo istotne jest, żeby w miarę możliwości wykonywać je równolegle, tym samym wykorzystując wszystkie dostępne zasoby CPU. Niepożądaną sytuacją jest np. zablokowanie wykonania innych prac w tle i zadań przez jedno, które zabiera zbyt wiele czasu. Takie zachowanie może prowadzić do stałego wzrostu kolejki żądań, tym samym blokując działanie systemu.

Ponadto zadania powinny być od siebie odizolowane na tyle, na ile to możliwe. Nie chcemy dopuścić do sytuacji, w której nieobsługiwany wyjątek w funkcji obsługi żądania doprowadza do awarii innego, niezwiązanego z nią wyjątku w funkcji obsługi żądania zadania w tle, a zwłaszcza całego serwera. Warto również unikać sytuacji, w której z awarią zadania wiąże się wystąpienie niespójności w pamięci, co może na dalszym etapie doprowadzić do błędu w innym zadaniu.

Dokładnie takie zadanie spełnia model współbieżności BEAM. Procesy pozwalają na wykonywanie czynności równolegle, osiągając tym samym *skalowalność* – zdolność do odpowiedzi na wzrost obciążenia poprzez wykorzystanie dostępnej mocy sprzętowej, z której system korzysta samoczynnie.

Procesy gwarantują również izolację, która z kolei zapewnia *odporność na błędy* – zdolność lokalizowania i ograniczania skutków nieuniknionych błędów czasu uruchomienia. Mogąc lokalizować wyjątki i naprawiać wynikające z nich błędy, będziesz mógł stworzyć system, który rzeczywiście nie przestaje pracować, nawet gdy wystąpią błędy.