Spis treści

1. Sponsorzy 11

2. Partnerzy projektu 13

3. O czym i dla kogo jest ta książka? 15

4. Skąd pomysł na napisanie książki? 18
   4.1. Co nowego w tym wydaniu? 20

5. Z jakich powodów nie warto zostać programistą? 21
   5.1. Dla pieniędzy 21
   5.2. Bo to „łatwa praca w biurze" 22
   5.3. Konieczność ciągłego rozwoju i śledzenia nowości 24
   5.4. A dlaczego warto? 25

6. Co powinieneś potrafić, by zacząć pracę jako programista? 28
   6.1. Język angielski 29
   6.2. Umiejętność szukania informacji 30
   6.3. Rozbijanie zadań na mniejsze i praca nad jedną rzeczą naraz 31

6.4. Napisanie (nawet bardzo małego) projektu od początku do końca 32
6.5. Umiejętność skonfigurowania środowiska pod projekt 34
6.6. Podstawy baz danych (zapisywanie i czytanie danych), technologii webowych, działania sieci, programowania obiektowego, pisania testów i webserwisów 35
6.7. Parsowanie danych JSON i XML 37
6.8. Praca z narzędziem kontroli wersji, np. Gitem 38
6.9. Usprawnianie pracy w IDE lub innym środowisku 39
6.10. Dla zielonych w programowaniu – od jakiego języka zacząć? 40
6.11. Już to potrafię – czy jestem senior developerem? 40

7. Szkoła, studia i odwieczne pytanie o matematykę 47
7.1. Bez matematyki ani rusz? 47
7.2. Jak podchodziłem do kwestii szkoły/studiów 48
7.3. Dlaczego warto iść na informatykę? 50
7.4. Dlaczego nie warto iść na informatykę? 51
7.5. Krótko o belfrach 53

8. CV, rozmowy kwalifikacyjne i szukanie pracy 54
8.1. Moje pierwsze CV na stanowisko programisty – dlaczego było do bani? 54
8.2. Drugie CV – lepsze, ale wciąż nieidealne 57
8.3. Gdybym dzisiaj przygotowywał CV 57

8.4. Proces szukania pracy 59
8.5. Rozmowy kwalifikacyjne 60
8.6. Zbieraj doświadczenie w aplikowaniu i wyciągaj wnioski 61
8.7. Zadanie praktyczne 62
8.8. Dlaczego nie polecam zaczynać od pracy zdalnej/freelancingu 63
8.9. Przydatne portale 64
8.10. Wątpisz w siebie, chociaż sporo wiesz? Prawdopodobnie masz syndrom oszusta 64
8.11. Podsumowanie 66

9. Dzień, organizacja i role w pracy. Agile, scrum i inne mądre słówka 67
9.1. Agile i scrum – z czym to się je? 68
9.2. Lepiej pracować według podejścia agile? 70
9.3. Sprint… 71
9.4. Role w organizacji pracy 74
9.5. Podsumowanie 76

10. Praktyczna ścieżka rozwoju – konkrety 77
10.1. Język angielski 77
10.2. Umiejętność szukania informacji 80
10.3. Dopisywanie „example" do szukanej frazy 80
10.4. Nie bój się szukać (nawet prostych rzeczy) 82
10.5. Szukaj i nabieraj doświadczenia 83
10.6. Krótki kurs zaawansowanego szukania w Google 84
10.7. Rozbijanie zadań na mniejsze i praca nad jedną rzeczą naraz 85

10.8. Karteczki samoprzylepne – popularne sticky notes ... 85
10.9. Trello – proste w obsłudze narzędzie do rozbijania zadań ... 88
10.10. Ćwiczenie skupiania się/wyciszania – krótko o medytacji (możesz olać ten punkt) ... 88
10.11. Napisanie (nawet bardzo małego) projektu od początku do końca ... 90
10.11.1. Manager schroniska dla zwierząt ... 91
10.11.2. Pierwsza gra ... 94
10.12. Skonfigurowanie środowiska pod projekt ... 97
10.13. Zagadnienia, w których powinieneś się orientować ... 100
10.13.1. Pliki z danymi, zapisywanie i czytanie danych ... 101
10.13.2. Bazy danych ... 103
10.13.3. Podstawowe technologie webowe ... 104
10.13.4. Protokoły komunikacyjne ... 106
10.13.5. Programowanie obiektowe ... 108
10.13.6. Web serwisy ... 111
10.13.7. Testy jednostkowe ... 114
10.13.8. Parsowanie danych JSON i XML ... 115
10.13.9. Dodawanie bibliotek i budowanie projektu ... 116
10.13.10. Praca z bugami i proces debugowania ... 117
10.13.11. Praca z narzędziem kontroli wersji, np. Gitem ... 118
10.14. Usprawnianie pracy w środowisku programistycznym ... 121
10.15. Podsumowanie ... 124

11. Z jakich źródeł się uczyć? Polecane strony /materiały 125
11.1. Najlepszy start w programowanie – świetna (i darmowa!) książka 126
11.2. Początki w Javie 127
11.3. Krótkie i przyjemne wprowadzenie do Ruby (po polsku) 131
11.4. Dobra książka do C++ 131
11.5. Źródła do nauki Pythona 132
11.6. Najlepsze źródło informacji o nowinkach w Androidzie – Android Weekly 134

12. Jak wygląda typowy tydzień pracy programisty? 135
12.1. Planowanie, estymacja, meetingi 135
12.2. Coś nagle nie działa i nie masz na to wpływu 137
12.3. Zależności ciąg dalszy 138
12.4. Bugfixing/debugging – szukanie i naprawianie błędów 139
12.5. Czasami wszystko idzie gładko! 141
12.6. Nauka w pracy 141
12.7. Krótkie podsumowanie typowego tygodnia 143

13. Jesteś gotowy do pierwszej pracy? Pytania 144

14. Pierwsze zadania programisty. Faktyczne zadania od czytelników 150

15. Co dalej? Parę słów na koniec 170