## Wrapping Up

Kotlin doesn’t want to be an average statically typed language, and it takes type safety to a whole new level. By setting nullable reference types apart from non-nullable types, the compiler introduces greater type safety but without introducing memory overhead. Kotlin also provides a number of operators to fetch objects from nullable references with great ease and fluency. The smart casts feature removes unnecessary casting wherever possible, which reduces noise in code. When working with generic functions and classes, you can enjoy type safety, along with greater flexibility, by tailoring the parametric type variance to meet your needs. Additionally, the reified type parameters reduce clutter and error in code by enhancing compile-time type safety when working with parametric types.

---

In the next part, we will dig into Kotlin’s support for object-oriented programming.