* Scala is a difficult language to master because its principles are based on mathematical type theory, which is fully understood by only the most academic and mathematically minded programmers. Furthermore, many language features of Scala, including implicits and macros, may cause the control of the program to flow unexpectedly to other parts of the code base, which makes it difficult for most programmers to follow or debug their code. An excellent programmer who comprehends all this will be more productive in Scala than in Java, but an average programmer’s productivity, as measured by implemented functionality, will probably decline when transitioning from Java to Scala. This is not just a short-term decline due to a learning curve — the decline has been observed in some development teams a ***full year*** after adopting Scala.

**Faster time to market**

If you are not an amateur developer without any constraints like the market of cost guiding you throughout the project, you may not find this argument interesting at all. But any paid developer has to deliver it on time. This is the particular area where Scala is incontestably better than Java. When it comes to Scala, the limitations of OO patterns for implementing l your code does not apply. Besides this pattern, the developers are also allowed to bring functional paradigms. This allows grabbing the best of both worlds, respectively functional coding paradigms and the OO patterns. What this ultimately results are very expressive detailed as well as concise codes with just a few lines of codes. Fewer lines of codes push the speed in every aspect, whether in testing and development.

When it comes to Scala, the limitations of OO patterns for implementing your code do not apply. In addition to this pattern, developers are also allowed to bring functional paradigms. This allows them to grab the best of both worlds, respectively functional coding paradigms and the OO patterns. This ultimately results in very expressive detailed as well as concise codes with just a few lines of codes. Fewer lines of codes will increase the speed in both testing and development.

**Options to avoid deadlock**

As a software developer, you might have experienced deadlock situations. The most dreadful null pointer exception or any other variants is a great example.

Can Scala offer a straightforward solution? No, but it offers several effective options to avoid the problem. When you start coding in Scala you are bound to observe a considerable decrease in the number of null pointer exceptions in your code.

**Better coding output**

First of all, functional programming will help your app get stability while flushing out all kinds of unintentional side effects. When you switch from the mutable data structures to so-called functional aspects, you ensure more safety, stability, and security.

Second of all, you can expect your code to be stronger without explicit punctuation. Moreover, the codes here support mixed capabilities and multiple inheritances. This will help you to identify any flaws in the code. The cleaner and stronger codes help the developer boost the app’s security and performance.

**Robust reserve of third-party APIs**

Every developer needs functionalities for their apps. Nowadays, when choosing the functions that you want in your app, you have two distinct options: build the function from scratch or accept help from a third party app. Now any program architect needs to cross this crucial juncture and decide either to build an entire function or just grab a plugin to take care of all these.

Not so fast! Scala offers a much more powerful solution. It comes loaded with built-in precious functions. Thanks to this you have far better controls. The simple and straightforward code will also boost the loading speed.

**Asynchronous processing**

Unlike Java, Scala is built to deliver for asynchronous behavior. The absence of future readiness in Java makes way for a lot of compromises that limit the functionality of the app. However, asynchronous behavior in Scala, just like other web development frameworks, offers extreme ease with regard to standout natural codes