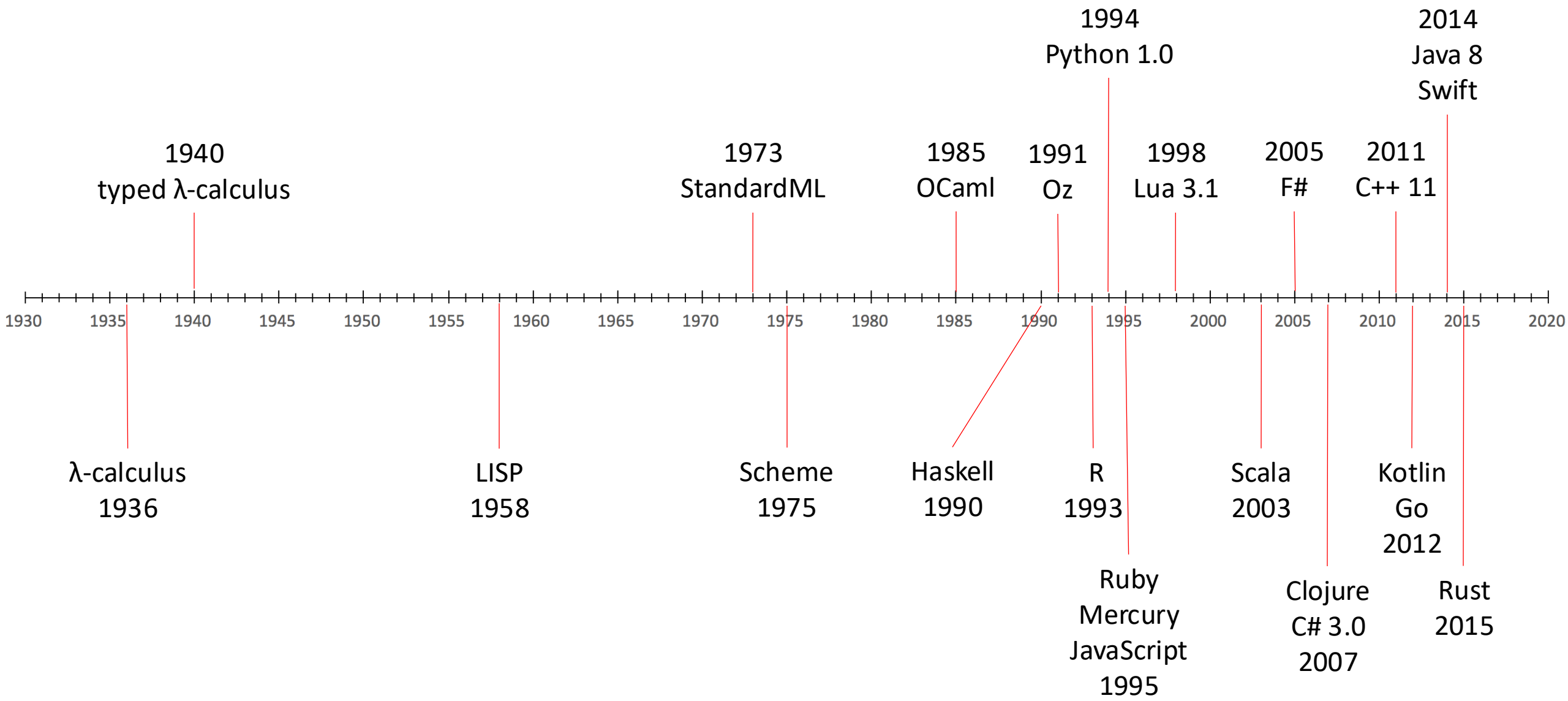
Year when lambda functions were introduced in various languages



data source: Sergei Winitzki's book: *The Science of Functional Programming: A tutorial, with examples in Scala*

slide by @philip_schwarz

Language	Year	Code for $k \mapsto k+1$
λ-calculus	1936	$\lambda k.\ add\ k\ 1$
typed λ-calculus	1940	$\lambda k : int.\ add\ k\ 1$
LISP	1958	`(lambda (k) (+ k 1))`
Standard ML	1973	`fn (k:int) => k + 1`
Scheme	1975	`(lambda (k) (+ k 1))`
OCaml	1985	`fun k -> k + 1`
Haskell	1990	`\ k -> k + 1`
Oz	1991	`fun {$ K} K + 1`
R	1993	`function(k) k + 1`
Python 1.0	1994	`lambda k: k + 1`
JavaScript	1995	`function(k) { return k + 1; }`
Mercury	1995	`func(K) = K + 1`
Ruby	1995	`lambda { |k| k + 1 }`
Lua 3.1	1998	`function(k) return k + 1 end`
Scala	2003	`(k:Int) => k + 1`
F#	2005	`fun (k:int) -> k + 1`
C# 3.0	2007	`delegate(int k) { return k + 1; }`
C++ 11	2011	`[] (int k) { return k + 1; }`
Go	2012	`func(k int) { return k + 1 }`
Kotlin	2012	`{ k:Int -> k + 1 }`
Swift	2014	`{ (k:int) -> int in return k + 1 }`
Java 8	2014	`(int k) -> k + 1`
Rust	2015	`|k:i32| k + 1`

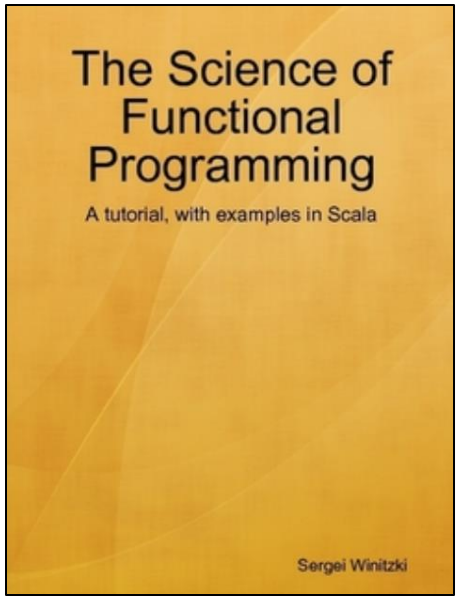
Table 1.2: Nameless functions in various programming languages.

1.7.7 Nameless functions: historical perspective

Nameless functions were first used in 1936 in a theoretical programming language called "λ-calculus". In that language,[9] all functions are nameless and have a single argument. The letter λ is a syntax separator denoting function arguments in nameless functions. For example, the nameless function $x \mapsto x+1$ could be written as $\lambda x.add\ x\ 1$ in λ-calculus, if it had a function add for adding integers (which it does not).

In most programming languages that were in use until around 1990, all functions required names. But by 2015, most languages added support for nameless functions, because programming in the map/reduce style (which invites frequent use of nameless functions) turned out to be immensely productive. Table 1.2 shows the year when nameless functions were introduced in each language.

What this book calls a "nameless function" is also called anonymous function, function expression, function literal, closure, lambda function, lambda expression, or just a "lambda". I use the term "nameless function" in this book because it is the most descriptive and unambiguous both in speech and in writing.



Sergei Winitzki

sergei-winitzki-11a6431