Go was conceived by Google because the languages they were using weren't able to solve multicore processors, networked systems, massive computation clusters, and the web programming model were being worked around rather than addressed head-on. Go works better on this environment than the other languages they were using (Java, Python, C, etc.). Go is a compiled, concurrent, garbage-collected, statically typed language developed, so it has the advantages of Python, C and Java. Another advantage is that Go can be used in servers using a distributed build system.

Go language had a lot of problems when it was launched, there were no so many information about the language, a few documentations, not so many programmers understood the language, etc. One of those problems was the block structure with braces, they had to change to brace bounded blocks.

These problems had been solved by practicing and fulfilling the documentation.

When Google created Go they focus on the following requirements:

- It must work on a large scale in terms of dependencies, team members.
- It must be familiar with the languages they were using: C is an example.
- It must be modern, be able to sustain on multicore machines, networking, and web application development. There are features of the modern world that are better met by newer approaches, such as built-in concurrency.

On dependencies, Go improved some technical approaches tan C: If the source file imports a package it does not use, the program will not compile. This guarantees by construction that the dependency tree for any Go program is precise.

Go compilation is faster than C or C++ compilation. There are two reasons for this. First, we found a bug: the Go compiler was generating a substantial amount of data in the export section that did not need to be there. Second, the export data uses a verbose encoding that could be improved.

Another feature of the Go dependency graph is that it has no cycles. The lack of circular imports causes occasional annoyance but keeps the tree clean, forcing a clear demarcation between packages.

Packages let an easier and more efficient (in libraries linking terms) for the compiler. You can also put remote packages, for example: a Github remote package.

The syntax has a Pascal reference, which has a more natural language comparing to C. Also the declaration of variables is more natural for new developers.

Identifiers follows a struct that guarantees scaling, clarity and robustness.

The semantics of Go statements is generally C-like. By design, it should feel familiar to programmers accustomed to languages in the C family.

Go language provides a concurrency environment for large scaling projects with multicore machines running web servers with multiple clients.

The garbage collector works with the philosophy of “If the programmer takes care to use memory wisely, the current implementation works well for production use.” It has an implementation similar as C and C++ garbage collector.

Interfaces are very alike as Java supporting its oriented object nature.

Go doesn’t have an exception facility in a conventional sense. In Go errors are just values and programs compute with them as they would compute with values of any other type. For this implementation, people prefer try-catch exceptions as Java.