Содержание

Многопоточность

Процесс и поток

- процессы, как правило, независимы
- процессы имеют отдельные адресные пространства, тогда как потоки выполнения совместно используют их адресное пространство
- процессы взаимодействуют только через предоставляемые системой механизмы связей между процессами

Многопоточность в Go

1. Высокоуровневая поддержка параллельного программирования в runtime
2. Go-рутины
3. GC

Go рутины

- независимо выполняемая функция, запущенная с помощью инструкции go
- У нее свой стэк
- Затраты по памяти ~ 2кб-4кб
- Это не поток
- Runtime Go занимается распределением горутин по тредам

Каналы

Каналы обеспечивают возможность общения нескольких горутин друг с другом, чтобы синхронизировать их выполнение.

По умолчанию каналы двунаправленные, но для улучшения читаемости и защиты от ошибок, можно явно указать тип канала как только принимающий или отправляющий.

Каналы

Небуферизованные

- Требуют синхронизации читателя и писателя, обе операции блокирующие

Буферизованные

- make(chan, cap), в канал можно отправить cap сообщений, до блокировки

runtime Go гарантирует окончание записи в канал, до начала чтения

Select

- Аналог switch для каналов
- Выбирает первый готовый канал, и получает сообщение из него, или же передает сообщение через него
- Если готовы несколько каналов, получение сообщения происходит из случайно выбранного готового канала
- Если не готов не один, то выполнение блокируется, или выполняется default case

sync

Mutex (RWMutex)

WaitGroup

Once