Go Game Server

Project GGS

En generisk spelserver i Go

Val av språk

- **Server** - Go eller Erlang?
- **Klient** - Java? C++? Python?
- Go
  - Imperativt.
  - Go-routines.
- Java
  - Stort standardbibliotek
  - Vi hade redan erfarenhet

Go VS Erlang

+ Imperativt
+ Effektivt
+ Spännande språk
- Ingen process supervision

Concurrency Model

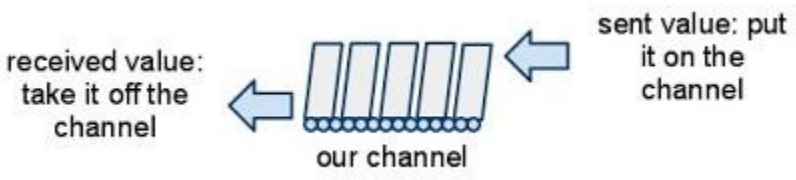
Go-routines

lättviktiga processer

Channels

Go-routines sätt att kommunicera

Concurrency Model

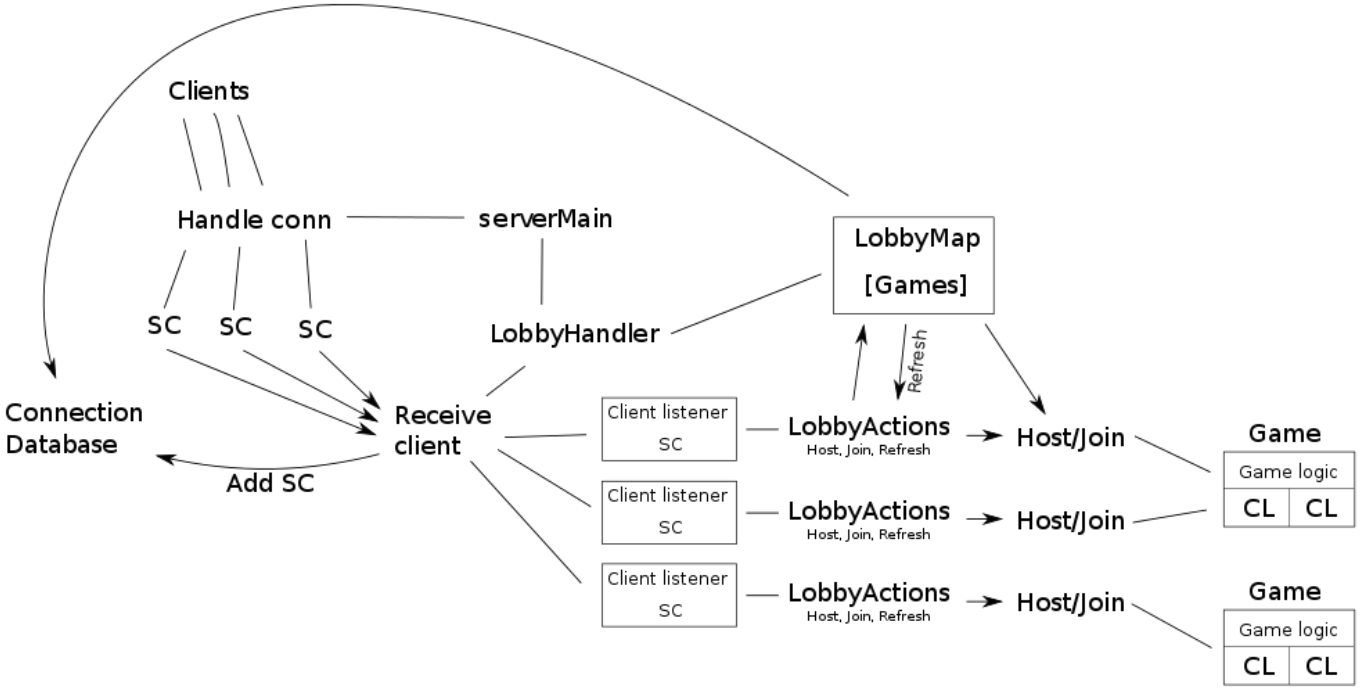



```
//a channel that can send and receive an int
ic := make(chan int)
```

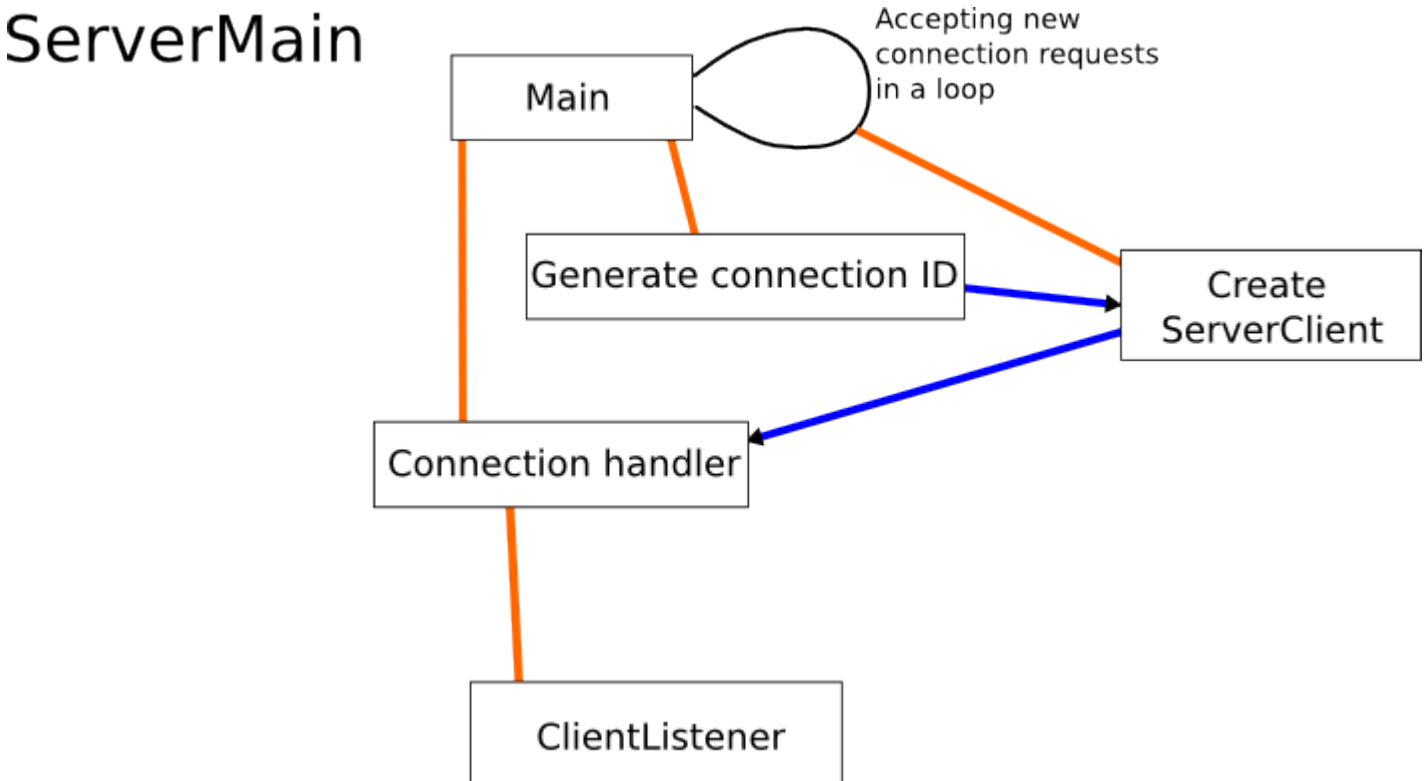
```
func add2Numbers(a, b int) {
    fmt.Println( a + b )
}

func main() {
 add2Numbers(1, 2) //a normal function call
 go add2Numbers(1, 2) //a normal function executed parallely as a goroutine
}
```

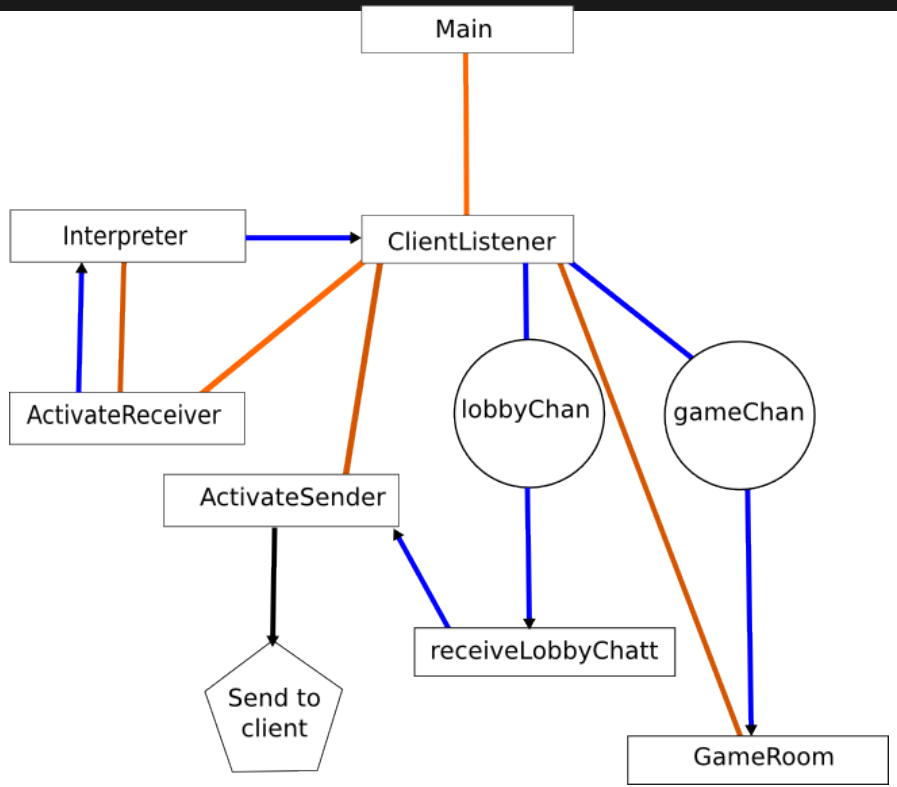
Systemarkitektur



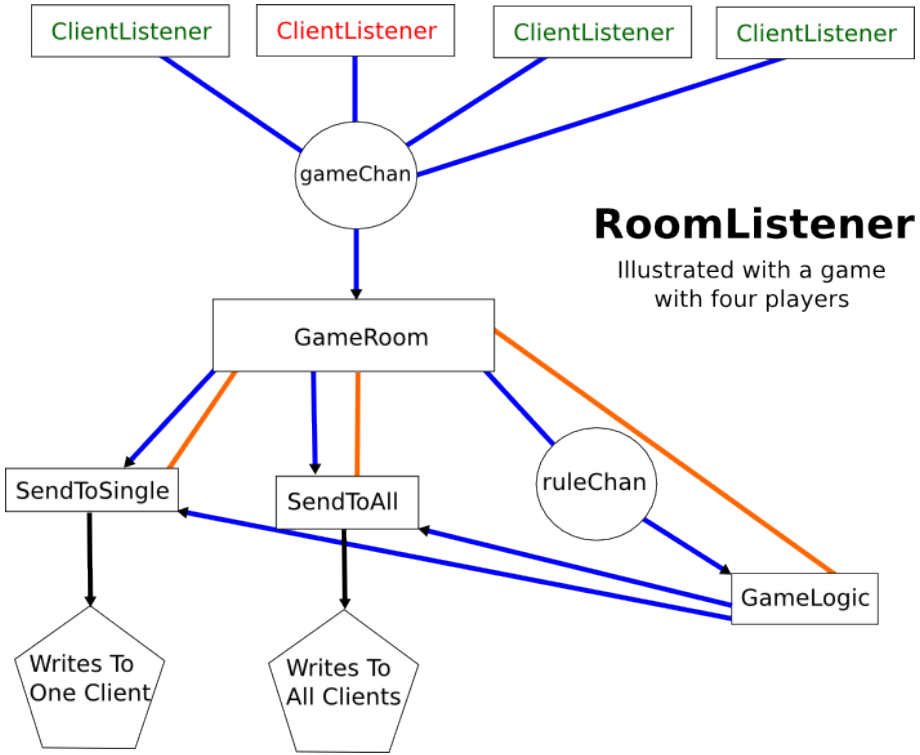
ServerMain


ServerMain
Main
Accepting new
connection requests
in a loop
Generate connection ID
Create
ServerClient
Connection handler
ClientListener

ClientListener



RoomListener



Concurrency och deadlock

- Vilka delar av system utnyttjar concurrency och vilka gör det inte?
- Behov av synkronisiering?
- För- och nackdelar med vald implementation?
- Alternativ till vald implementation?

Demo

Organisation och Planering

- Arbetat tillsammans
- Serverlogik - Klient & Spellogik

Bra och dåligt

+ Databas, kan hantera alla datatyper
+ Helheten
+ Model-View-Controller
- Abstraktion
- Generiskt