ICA 06 gruppe 2

ICA 06 har vi tatt kode fra gruppe 11. I denne koden har de brukt «main package».

Packages:

I denne «main package» finner vi disse importerte pakkene: Bytes, fmt, io, martini, net/http og net/URL

Bytes: en pakke som implementere funksjoner for å manipulere byte slices.

Fmt: Pakken gir oss flere muligheter for å formatere ulike Og verdier, dette gjøres ved hjelp av utskrifts «verb». F.eks %x som brukes for å «encode» hexa-desimaler.

IO: Pakken gir grunnleggende grensesnitt til io-primitiv. Grensesnittet i io er hovedsakelig «Reader» og «Writer» diss to grensesnittene er til for å støtte «Read» og «Write» metoder. Det er mange funksjoner i io som bruker en av disse grensesnittene som argumenter.

Net/http: pakken inneholder implementasjon for http klient og server.

Net/URL: Pakkeadresse som analysere nettadresser og implementerer QuerryEscaping

API:

Det er deretter definert en variabel men navnet «BaseURL», denne variabelen inneholder en API til google-translate.

Funksjoner:

Func main:

Ved å skrive m := martini.classic() har de definert bokstaven m som navnet til martini.classic(). Når de da kaller på m.Get så kaller de get funksjonen til martini.

Ved martini pakken som blir brukt finner vi ut av at de bruker martiniparams, fra funksjonen finner vi ut at alt som kommer etter «speech/:» er en parameter. På linje 22 kan vi se at teksten blir hentet fra parameteren.

Linje 23: speech: gjør at alt som kommer inn i parametere tekst blir gjort om til tale, vi får også muligheten til å endre talespråk.

I *linje 25*: defineres innholdstypen i URL til audio/mpeg.

I *linje 27*: ser vi at speech skriver til http.responswritter, deretter vil den returnere en audio.

m.runOnAddr(«:8080»): Ved hjelp av denne koden kan vi definere en ønsket port.

Fra *linje 33*, lages det et struct(collection of fields) av type speech. Funksjonen Byte.buffer gjør at den «slicen» av data blir sendt opp til buffer, bufferen vil da være tilgjengelig fem til en ny request blir gjort.

Linje 38 til 51 har vi en funksjon som henter lyd fra google-translate.

Ved hjelp av sprintf, kan vi formaterer BaseURL til tekst og språk. Dette gjøre ved å bruke queryescape(queryescape som også kan kalles for URL encoding, data som blir overført til webapplikasjoner er nøtt til å bli «encoded», dette gjøres ved hjelp av url.QueryEscape). Vi ser at det i dette tilfelle er brukt QueryEscape til å kode parlamentærene text og language .

Linje 41: her sjekkes det for evt. Feil og gir tilbakemelding hvis feil.

Linje 50: Her brukes et http-get metode for å hente variabelen «req».

Linje 45 til 50: Det er en speech metode, vi bruker en if settning som sender en feilmelding til buffer hvis det er en error. Leser responsbody kopier den til buffer og sjekker deretter for feil. Hvis feil er funnet returneres det en feilmelding.