General

Collapse all

Rules for the exam (see the attached PDF for details)

- The exam consists of **two parts**: the programming project and the oral exam. Both parts are required to pass the exam.
- Only students who obtain a sufficient mark in the programming project are admitted to the oral exam.
- Every plagiarism on the project will be severely punished according to UNIFI rules.
- The mark assigned by the teacher is not "negotiable", meaning that the mark cannot be increased by taking an additional oral test or completing an extra assignment.
- If a student fails the exam, he/she cannot retake it before the next exam date.
- Students must refrain from mentioning their personal circumstances to influence the exam result.
- The students are strongly encouraged not to request exceptions to the above rules.

Annunci

GitHub repository with programming exercises

Website where to download the textbook "M. van Steen and A.S. Tanenbaum, Distributed Systems, 4th ed."

Rules for the exam and the final project

Introduction to Distributed Systems and Design Goals

Introduction to distributed systems

- Examples of distributed systems
- Definition of a distributed system
- Description of the main design goals
  - Resources sharing
  - Distribution transparency
  - Openness
  - Scalability
  - Availability
  - Modularity

Material to study for the exam

- Chapter 1 of "*Distributed Systems*" by M. van Steen and A. S. Tanenbaum

Introduction to the course

Introduction to distributed systems

Entrance Test

Answers to the Entrance Test

?

Introduction to Go Programming

Programming in Go

- Packages, variables, and functions
- Flow control statements
- Complex types: struct, slices, and map
- Methods and interfaces
- Generics
- Concurrency

Case study

- Go Pong (see the attached code)

Material to study for the exam

- A tour of Go
- Go Bookcamp
- Learn Go
- Go Documentation
- Go By Examples
- Go with Visual Studio Code
- How to write Go code
- Organizing a Go module
- Comprehensive Guide to Testing in Go

 Case study: Go Pong

Distributed System Architecture

Software architectures

- Layered organization
- Distributed objects
- SOA
- REST
- Publish-subscribe
- Tuple spaces

System architectures

- Client-server
- Peer-to-peer
- Cloud
- Edge
- Blockchain systems

Case study

- Bitcoin

Material to study for the exam

- Chapter 2 of "*Distributed Systems*" by M. van Steen and A. S. Tanenbaum

 Distributed System Organization

Principles of Blockchain Systems

Case study: Bitcoin

Network Programming in Go

Marshal/Unmarshal data structure in Go

- Marshal and unmarshal data in JSON
- Marshal and unmarshal data in XML
- Marshal and unmarshal data in YAML

Network programming

- Create a TCP client and server
- Create a UDP client and server
- Perform HTTP requests

Case studies

- Todo list app (see attached code) --- it uses dependency injection
- TCP Chat client and server (see attached code)

Material to study for the exam

- JSON and Go
- How to Use JSON in Go
- Dependency Injection in Go

Marshaling and unmarshaling of data

Dependency Injection

Network programming

Case study: Todo list app

Case study: gochat

Examples: client and server TCP, UDP, and HTTP client

Web Programming in Go

A first of Programming Web Application

- Review of HTTP protocol
- Structure of a Web applications
- Examples of a first web application in Go

Handling and processing requests

- Request handlers, function handlers, and middleware
- Managing requests parameters
- Prepare responses
- Managing cookies

Storing and accessing data

- Connection to databases
- Create tables and insert data
- Performing queries
- GORM relational mapper

Display content

- Creating and parsing templates
- Go Template actions
- Layouts and static files
- Examples of complete web applications

Material to study for the exam

- Writing Web Applications
- Server-side Web Development
- Building Modular and Testable Web Application
- A complete guide to working with Cookies in Go
- How to manage configuration settings in Go web applications
- Go database/sql tutorial
- SQLite tutorial
- Database Operations in Go using GORM
- GORM doc


Introduction to Web Programming in Go


GoWiki: a simple example of web application


Handling and processing requests


Examples: requests handlers, parameters, and cookies


Storing and accessing data in a database


Examples on databases access


Display content using Web Templates


Examples on display content using Web Templates


Example of Web Application: GoForum

Communication Mechanisms

Remote Procedure Call

- Basic request-replay model
- Approaches to pass parameters to remote procedure
  - Serialization
  - Global/local references
- Asynchronous RPC
- Multicast RPC

Message-oriented Middleware

- Queuing model
- Communication properties
- Queue Managers
- AMQP Protocol
- ZeroMQ Framework

Material to study for the exam

- Chapter 4 of "*Distributed Systems*" by M. van Steen and A. S. Tanenbaum
- RabbitMQ Tutorials 1 and 3
- Chapter 1 of "ZeroMQ - The Guide"

Communication mechanisms

Examples of using ZeroMQ Framework in Go

Coordination and synchronization mechanisms

Coordination and synchronization Mechanisms

- Time synchronization protocols
- Lamport's Logical clocks
- Vector clocks
- Mutual exclusion protocols

Distributed Algorithms

- Broadcast problem: flooding algorithm
- Spanning Tree construction problem: SHOUT protocol
- Leader election in static networks: yo-yo protocol
- Leader election in dynamic networks: Bully-algorithm and election in Ad Hoc Networks

Material to study for the exam

- Chapter 5 of "Distributed Systems" by M. van Steen and A. S Tanenbaum
- Design and Analysis of a Leader Election Algorithm for Mobile Ad Hoc Networks
- Notes given by the teacher (see files below)

Coordination and synchronization

Leader Election in Dynamic Network

Leader Election in Dynamic Network

Broadcast problem

Spanning Tree Construction

Leader election and yoyo protocol

Notes on distributed algorithms

Web Services in Go

RESTful services

- Write a client to interact with a REST service
- Imlementation of a CRUD service
- Example of service

gRPC services

- Define a service using ProtocolBuffer
- Implement a remote service
- Using a remote service
- Streaming interaction between client and server

Material to study for the exam

- Introduction to gRPC
- gRPC: Core concepts, architecture and lifecycle
- gRPC in Go: quick start and basic tutorial

RESTful Web Services in Go

gRPC in Go

Examples of RESTFul Web Services

Examples of gRPC Web Services

Go Polling REST Service

Internet of Things

Pervasive systems

- Ubiquitous computing
- Mobile computing
- Sensor networks

Internet of Things

- IoT Architecture
- IoT Protocol stack

Application-level IoT Protocols

- MQTT protocol
- Programming MQTT publishers and subscribers in Go through Eclipse Paho library
- COAP protocol
- Programming CoAP resources and clients in Go through go-coap library

Material to study for the exam

- Chapter 1 of "*Distributed Systems*" by M. van Steen and A. S. Tanenbaum
- MQTT Essentials
- How to use MQTT in Golang
- The Constrained Application Protocol (CoAP) RFC 7252 (sections 1 and 2)

Pervasive Systems and IoT

MQTT Protocol

Coap Protocol

MQTT in Go

Coap in Go

Examples of MQTT in Go

Examples of Coap in Go