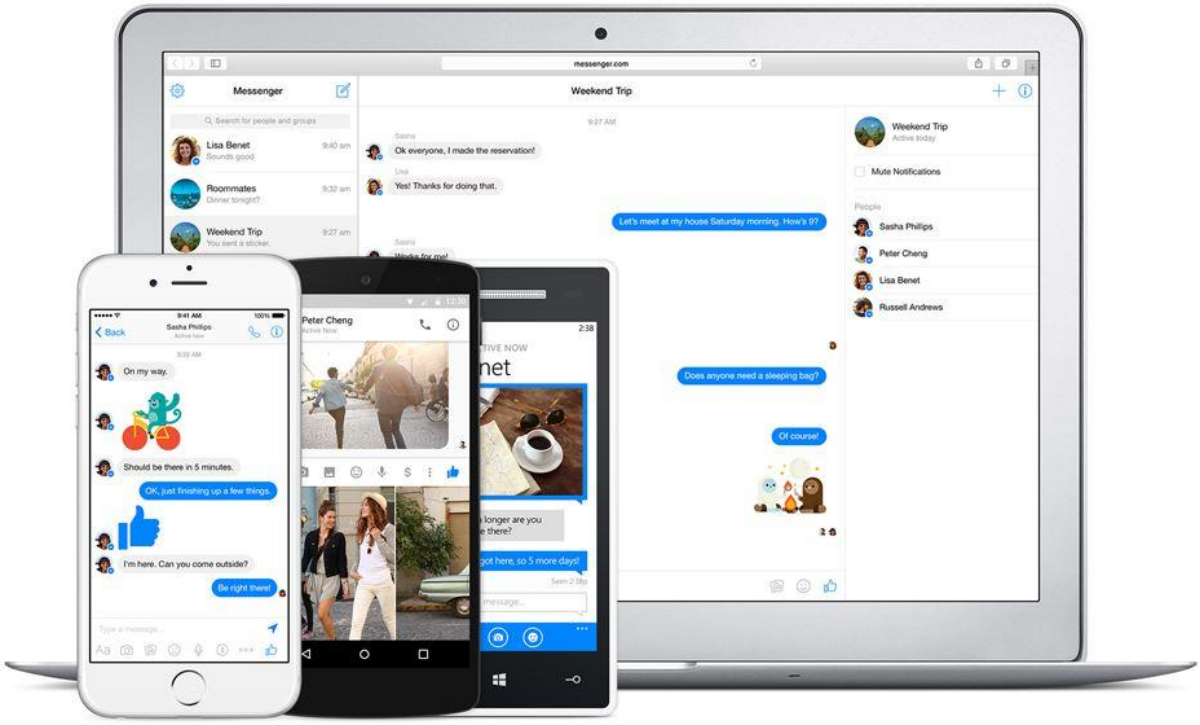
TDDD49/725G66 - Programmering i C# och .NET

LABORATIONER

Linköpings universitet



Inledning

Temat för denna laborationsserie är en peer-to-peer (P2P) chat-app för Windows. Chat-appen ska göra det möjligt att två användare ska kunna chatta med varandra. Användaren ska kunna skicka chattinbjudan och vänta på att andra parten accepterar den. Sedan kan båda användare skicka textmeddelanden och bilder till varandra. Hela kommunikationen ska vara baserad på portnummer och ip-adress. Användare ska INTE behöva skapa något konto och chat-appen ska vara serverlös.

Tänk på att labbarna hänger ihop. När du löser en labb, gör det med de andra labbarna i åtanke. Detta gäller särskilt den första labben som också kräver mer planering.

Några P2P chat-appar:
https://www.slant.co/topics/7254/~cross-platform-secure-p2p-chat-application#1

Tox
https://www.youtube.com/watch?v=kA3jtFTFm5I

Din implementation ska bara följa kraven och ovanstående nämnda apparna är bara exempel på hur en P2P chatt-app kan se ut. Det finns inget krav att din implementation ska se exakt likadant ut. Med det sagt, rekommenderar vi er att kolla på dem nämnda apparna som en inspirationskälla.

LABB 1

Syfte

I slutet av labben förväntas du ha implementerat nedanstående funktionalitet. Målet med labben är att möjliggöra en anslutning mellan två chattklienter.

Funktionella Krav

- Användaren ska kunna skicka inbjudan till en annan användare genom att använda IP-adress och portnummer. Om det inte finns någon annan användare som lyssnar på porten då ska ett felmeddelande visas för användaren som har skickat inbjudan.
- Användaren ska kunna börja lyssna på ett angivet portnummer, acceptera eller neka inbjudan. När inbjudan nekas eller accepteras ska användaren som har skickat inbjudan få återkoppling.
- Användaren ska även kunna se när den hen chattar med avslutar anslutningen eller när anslutningen avbryts av någon annan anledning.
- Varje chattmedlem i en anslutning ska ha ett namn, som inte behöver vara unikt, som anges av användaren när anslutningen skapas.

⚠ ***Det är INTE tillåtet att använda Windows Forms i gränssnittet.***

⚠ ***Läs de icke-funktionella kraven (finns i slutet av dokumentet) innan du börjar med labben. Det är jätteviktigt att alla labbar följer dessa.***

⚠ ***Man får använda 127.0.0.1 ip-adressen för att skapa anslutning mellan två olika instanser av sin implementation på samma maskin.***

LABB 2

Syfte

I slutet av labben förväntas du ha implementerat nedanstående funktionalitet. Målet med labben är att implementera en textbaserad chatt mellan två chattklienter.

Funktionella Krav

- Efter att chattinbjudan har accepterats ska användaren kunna starta en textbaserad konversation med den som hen är ansluten till.
- Båda medlemmarna i en konversation ska kunna skicka textmeddelanden till den andra medlemmen.
- Om en användare avbryter anslutningen/konversationen på något sätt ska den andra chattmedlemmen få återkoppling.

Icke-funktionella Krav

- `Panel` och `ObservableCollection` ska användas för att visa meddelanden i konversationer.

Det grafiska gränssnittet ska inte blockeras medan klienten väntar på svar eller skickar data till nätverket.

Det är inte tillåtet att använda TextArea för att visa konversationer.

⚠ ***Kolla de icke-funktionella kraven (sist i dokumentet) innan du kommer igång med labben. Det är jätteviktigt att er lösning också uppfyller dem.***

LABB 3

Syfte

I slutet av labben förväntas att du ha implementerat nedanstående funktionalitet. Målet med labben är att implementera en lokal filbaserad databas för att kunna spara konversationer. Konversationerna ska sedan kunna öppnas och visas i det grafiska gränssnittet.

Funktionella Krav

- Användaren ska kunna se en historik på alla sina konversationer med andra användare som hen har chattat med tidigare, även efter att appen har stängts ner och startats upp igen. Samma användarnamn kan förekomma i flera konversationer.
- Konversationer ska visas sorterat baserat på datum och tid.
- Användaren ska kunna välja och se en konversation från historiken.
- Man ska kunna söka efter konversationer man har haft med en person genom att ange hela eller en del av personens namn. Sökningen ska implementeras med LINQ.

⚠ ***XML eller JSON ska användas för att lagra konversationer lokalt.***

⚠ ***Kolla de icke-funktionella kraven (sist i dokumentet) innan du kommer igång med labben. Det är jätteviktigt att er lösning också uppfyller dem.***

LABB 4

Syfte

I slutet av labben förväntas du ha implementerat nedanstående funktionalitet. Målet med labben är att implementera stöd för att skicka och lagra bilder i konversationen.

Funktionella Krav

- Användaren ska kunna skicka “Buzz” till andra parten. Detta ska leda till att fönstret på mottagande parten skakar eller/och spelar något ljud.
- ***(Icke-obligatoriskt)*** Användaren ska kunna skicka bilder till den andra medlemmen i konversationen. Bilderna ska lagras lokalt och visas i konversationen.
- ***(Icke-obligatoriskt)*** *Användaren ska kunna ändra följande grafiska inställningar på chattfönster lokalt:*
    - *Fontstorlek*
    - *Fonttyp*
    - *Backgrundsfärg*

Varje chattfönster ska ha egna separata inställningar som lagras lokalt.

Kolla de icke-funktionella kraven (sist i dokumentet) innan du kommer igång med labben. Det är jätteviktigt att er lösning också uppfyller dem.

Icke-funktionella krav

Din implementation ska uppfylla följande krav:

- **Felhantering(Icke-obligatoriskt):**
  Ni ska använda er felhantering i implementationen. Vid slutet av varje labb ska följande gälla:
  - Don't throw new Exception()
  - Don't put important exception information on the Message field
  - Don't catch (Exception) more than once per thread
  - Don't ever swallow exceptions
  - Cleanup code should be put in finally blocks
  - Don't return special values on error conditions
  - Don't use exceptions to indicate absence of a resource
  - Don't use exception handling as means of returning information from a method
  - Don't clear the stack trace when re-throwing an exception
  - Avoid changing exceptions without adding semantic value
  - Each exception class should have at least the three original constructors

  Läs Exception Handling Best Practices in .NET för att veta mer om ovanstående kraven.

  ⚠ Det ska finnas minst ett eget definierat exception som skapats och använts i den färdiga chattapplikationen.

  Användaren ska få återkoppling på vad som har gått fel när det passar.

- **Nätverk**
  System.Net.Sockets namespace ska användas för att implementera P2P kommunikation.
  https://docs.microsoft.com/en-us/dotnet/api/system.net.sockets?view=netframework-4.7.2

- **Trådar**
  Trådar ska användas om det behövs. Det är viktigt att det grafiska gränssnittet inte blockeras medan klienten skickar data över nätverk eller väntar på data.
  https://docs.microsoft.com/en-us/dotnet/standard/threading/managed-threading-basics

  Det är också tillåtet att använda Task.
  https://docs.microsoft.com/en-us/dotnet/standard/parallel-programming/task-based-asynchronous-programming

- **WPF**
  Windows Presentation Foundation(WPF) ska användas för att bygga GUI:n..
  https://docs.microsoft.com/en-us/dotnet/framework/wpf/getting-started/introduction-to-wpf-in-vs

  - **MVVM**
    MVVM ska användas för att strukturera koden.
    https://www.wintellect.com/model-view-viewmodel-mvvm-explained/

  - **ICommand**
    ICommand interface ska användas lämpligt i implementationen av GUI.
    https://www.codeproject.com/Articles/1052346/ICommand-Interface-in-WPF

  - **Databinding**
    Databinding ska användas för att uppdatera gränssnittet eller läsa in inmatningar.
    https://docs.microsoft.com/en-us/dotnet/framework/wpf/data/data-binding-overview

- **Protokoll**
  Ni ska skapa eget protokoll för att utbyta data mellan två användare. Ni ska använda er json för datautbyte. Till exempel:

```
{
      "requesttype": "establishconnection",
      "name”: "Wonderer",
      "date": "2018-10-22T08:31:00.000Z"
}
```

Vid redovisning ska du motivera och förklara dina (tekniska) designval - varför klassindelningen ser ut som den gör, varför delarna av din applikation interagerar med varandra på de sätt de gör med mera.

Lycka till!