

**İSTANBUL TEKNİK ÜNİVERSİTESİ  
ELEKTRİK-ELEKTRONİK FAKÜLTESİ**

**MOBİL CHAT UYGULAMASI**

**Bitirme Ödevi**

**Hilal AKAY  
040020705**

**Bölüm : Bilgisayar Mühendisliği  
Anabilim Dalı: Bilgisayar Bilimleri**

**Danışman : Yrd. Doç. Feza BUZLUCA**

Mayıs 2005

## **Özgünlük Bildirisi**

1. Bu çalışmada, başka kaynaklardan yapılan tüm alıntıların, ilgili kaynaklar referans gösterilerek açıkça belirtildiğini,
2. Alıntılar dışındaki bölümlerin, özellikle projenin ana konusunu oluşturan teorik çalışmaların ve yazılım/donanımın benim tarafımdan yapıldığını bildiririm.

İstanbul, 13.05.2005

Hilal AKAY

## ÖNSÖZ

Bitirme projem sırasında bana yol gösteren ve yardımcı olan Sayın Yrd. Doç. Feza Buzluca Hocama saygı ve teşekkürlerimi sunarım.

Proje boyunca birlikte çalıştığım MOBINEX firmasına katkı ve destekleri için teşekkür ederim.

Hayatım boyunca bana daima destek olan Aileme, özellikle başarımın en büyük kaynağı olan Elif Akay' a sonsuz saygı, sevgi ve şükranlarımı bir borç bilirim.

# MOBİL CHAT UYGULAMASI

## ( ÖZET )

Mobil uygulamalar ve kablosuz iletişim, teknolojik gelişmelerin en fazla ve hızlı yaşandığı alanlardır. Hem iş hem de eğlence amaçlı, çok çeşitli mobil uygulamalar kullanılmaktadır. Mobil cihazlarda, taşınabilir olmanın getirdiği kısıtlamalar, ağ bağlantısı ile giderilebilir. Kablosuz bağlantılar işlere esneklik ve güç kazandırır. Mobil uygulamaların getirdiği avantajlar ve gelişime açık olması, çok çeşitli mobil cihazların üretimini de beraberinde getirmiştir. Özellikle cep telefonları, değişik üreticiler tarafından, farklı standartlarda üretilmektedir. Bu durum uygulama geliştiricilere cep telefonu gibi kısıtlı cihazlarda program yazmanın güçlüklerinin yanında çok değişik özelliklere sahip bu cihazlara spesifik programlar yazma zorunluluğunu da beraberinde getirmiştir.

J2ME (Java 2 Micro Edition), mobil cihaz programlamasında en çok kullanılan dillerden biridir ve Java'nın, aynı uygulamanın hiçbir değişikliğe uğratılmadan farklı platformlarda çalışabilmesi özelliği sayesinde bir çok cep telefonunda çalışabilecek uygulamalar yazılmasında kullanılmaktadır. Fakat cep telefonlarının farklı ekran boyutları, tuş takımları ve bellek kısıtları, yazılan uygulamanın farklı cihazlarda bambaşka şekillerde görünmesi ve çalışmasına sebep olmaktadır.

Bu projede hedeflenen, değişik özelliklere sahip cep telefonlarında çalışabilecek bir mesajlaşma uygulaması yazmaktır. Projenin amaçları aşağıdaki gibi verilebilir:

- Değişik ekran büyüklüklerine, bellek kısıtlamalarına ve tuş kombinasyonlarına sahip cihazların tümünde aynı şekilde görüntülenecek arayüz bileşenlerinin oluşturulması, arayüz bileşenlerinin yönetiminin sağlanması
- Cep telefonundan sunucu bağlantısının gerçekleşmesi ve sunucu ile mesajlaşmanın sağlanması (Sunucu olarak MOBİNEX firmasına ait POC+ sunucusu kullanılmıştır)

Proje sonucunda, değişik cep telefonu modellerinde çalışabilen bir chat uygulaması oluşturulmuştur. Ayrıca, oluşturulan arayüz elemanları değişik türdeki bir çok uygulamanın yazılmasına olanak vermektedir. Oluşturulan bileşenler, cihazlar arasındaki farklılıkları azaltmakta ve uygulama geliştirmeyi kolaylaştırmaktadır. Oluşturulan chat uygulaması, farklı cihazlara sahip bireylerin uzak mesafelerden kolay haberleşmesini sağlamaktadır.

# MOBILE CHAT APPLICATION

## ( SUMMARY )

Mobile applications and wireless communication are the two main areas in which technological progress is the fastest and the most excessive. Many kinds of mobile applications are used for either business or entertainment. The limitations of the mobile devices can be overcome by networking. Wireless networks supply flexibility and power to mobile devices. The advantages of the mobile applications and the fact that it is open to development resulted in the production of a variety of mobile devices. Especially mobile phones are produced with different standards by different vendors. The wireless industry has a speed at which a technology runs through its life cycle. A brand new product can become old within months due to fast improvement of technology. That rapid change causes the lack of standardization between the devices in the wireless world. Thus, the application developers must face not only the difficulty of writing programs for devices with limited sources but also writing specific programs for these devices who have different properties.

J2ME (Java 2 Micro Edition), is one of the most commonly used programming languages in mobile device programming and as a result of the features of Java which enable platform-independent program development, it proves to write applications that can run in different mobile phones. But, because of the variances in the screen sizes, key pads and memory limitations of the mobile phones, the applications may appear and run differently in different devices.

The aim of this project is to develop a mobile messaging application, which can run in a variety of mobile phones with different properties. The main parts of the project can be listed as below:

- ❑ Developing user interface components, which can be viewed in the same form in different mobile phones with different screen sizes, keypads and memory limitations and management of these components.
- ❑ Implementing the sever-mobile phone client connection and enabling the device to communicate with the server. (The POC+ server of the MOBINEX company is used)

Java 2 Micro Edition (J2ME) is Sun's version of Java aimed at machines with limited hardware resources such as PDAs, cell phones, and other consumer electronic and embedded devices. The virtual machine specified in any profile is not necessarily the same as the virtual machine used in Java 2 Standard Edition (J2SE) and Java 2 Enterprise Edition (J2EE).

J2ME is a collection of technologies and specifications that are designed for different parts of the small device market. Because J2ME targets such a variety of devices, it is divided into *configurations*, *profiles*, and *optional packages*.

A J2ME configuration defines a minimum Java platform for a family of devices. A configuration can cover a broad range of devices. A configuration specification is aimed at devices with similar requirements of memory size and processing power. To ensure portability across the device range, configurations cannot contain any optional features. Configurations details a base set of APIs that can be used with a certain class of device. Limitations of the CLDC (Connected limited device configuration) are:

- ◆ 160KB to 512KB total memory available for Java technology
- ◆ Limited power (often battery)
- ◆ Limited, connectivity to a network
- ◆ Extremely constrained user interfaces, small screens

Profile is a collection of Java technology APIs that supplement a configuration to provide capabilities for a specific device or market type. A profile builds on a configuration but adds more specific APIs to make a complete environment for building applications. While a configuration describes a JVM (Java Virtual Machine) and a basic set of APIs, it does not by itself specify enough detail to enable you to build complete applications. Profiles usually include APIs for application life cycle, user interface, and persistent storage. An optional package provides functionality that may not be associated with a specific configuration or profile. This optional package could be implemented alongside virtually any combination of configurations and profiles.

In this project, MIDP2.0 devices with CLDC 1.0 and upper are supported. This choice enables the use of Socket programming in the mobile phones, which is used in the chat application.

The programs that are written for MIDP profile are called Midlets. One or more midlets can be packaged in a JAR file. Each JAR file has a descriptor in the form of a JAD file that describes what is inside the JAR file. The Application Management Software uses these files. The jar file contains the classes and resources (like pictures). A midlet can be installed to a mobile phone via computer or Internet with downloading these jad and jar files.

MIDP user interface consists of two main parts: low-level user interface components known as *Canvas* and high-level user interface components which are *Alert*, *Form*, *List* and *Textbox*. The aim of these high-level objects is to provide the portability of Midlets between different devices. The appearance of these components can change according to the devices' user interface. On the other hand, low-level user interface component provides direct control on the user interface. For instance, it provides a wide control on what to be drawn on the screen and how key events will be handled. The midlets can be working in different devices so some controls must be done according to device properties like the screen size, colors of the screens in order to make Midlets adaptable to different platforms. Besides, the user-interface has a menu object and is assigned to specific keys of the phone known as *RSK* (*Right Soft Key*), *LSK* (*Left Soft Key*) and *MSK* (*Midle Soft Key*). The appearance of this menu also changes in different phones. Also different actions can be

assigned for different keys in devices. By using Canvas, the programmer gains full control over all these actions.

In the project a **GUI package** is developed for the user interface components. In this package classes are created for main user interface components. While designing this package, it is considered to make these components usable for different applications. The main interface components are:

*FocusableItem*: The objects that can be focused extends from this class, implementing specific actions like *onFocus()* and *lostFocus()*

*ImageItem*: Used for showing the image on the screen. Different devices support different types of image formats. But according to MIDP profile, the standard image format is PNG, which must be supported by all MIDP devices.

*Label*: Used for showing strings on screen.

*Edit*: Used for user input.

*Button*: Used for activation of events when pressed on.

*Menu*: A collection of options listed. It is assigned to LSK or RSK of the device. It is consisted of a collection of Labels. The options in the menu can be arranged dynamically (adding or removing options on runtime). The most important part of the menu is the compatibility with different device sizes.

*Page*: A container class for holding all these objects on. It can be thought as a screen object.

It determines the design of the screen and management of the items on it. A page object is created for each screen design. The pages can be either pop-up dialog or full screen. *WaitingDialog* is a specific kind of page used for alerting the user to wait.

*List*: Used for holding different types of interface components together in an order and provides navigation between them. It consists of elements that implement the *ListElement* interface.

*GUIManager*: Holds the pages of a program and manages them. It is extended from the Canvas class of Java (the low-level user interface components). It mainly forms the screen appearance according to the page active at that time and provides key management.

*Actions*: The interactions of the objects explained above are provided by the actions. There are different kinds of actions like Sending SMS, opening or closing a page, showing the menu or changing an object attribute. Actions are added to the events like *onShow*, *onHide* and *onPressed*.

A mobile phone's value comes in its inherent wireless networking capabilities. One of the most critical aspects of J2ME is network connectivity. Many J2ME devices support to establish a wireless network connection. J2ME applications, in this regard, become more

than simple communication devices. They become another client capable of interfacing with the enterprise systems, databases, corporate intranets and the Internet.

The variations in J2ME devices make it difficult to design networking facilities for CLDC-based devices. Defining a new set of classes for I/O and network connectivity solves this problem. These classes are known as the generic connection framework. The idea of the Generic Connection framework is to define the abstractions of the networking and file I/O as general as possible to support a broad range of handheld devices, and leave the actual implementations of these abstractions to individual device manufacturers. These abstractions are defined as Java interfaces. The device manufacturers choose which one to implement in their MIDP based on the actual device capabilities. The GCF provides a single set of related abstractions you can use at the programming level to handle multiple forms of communications, instead of using different abstractions for different protocols. This platform-independent framework provides its functionality without dependence on specific features of a device. It provides a hierarchy of connectivity interfaces, but it does not implement any of them. Implementations are to be provided by profiles (such as MIDP). There is no implementation of the connection interfaces at the CLDC level. The actual implementation is left to MIDP.

Version 1.0 of the Mobile Information Device Profile (MIDP) lacks low-level networking support for TCP/IP sockets and UDP/IP datagrams, but the MIDP 2.0 specification (JSR 118) has responded to the needs of the 2.5G and 3G networks now being deployed by adding support for sockets and datagrams, thus providing mobile applications more capable networking interfaces.

TCP provides a reliable, point-to-point communication channel that client-server application on the Internet use to communicate with each other. To communicate over TCP, a client program and a server program establish a connection to one another. Each program binds a socket to its end of the connection. To communicate, the client and the server each reads from and writes to the socket bound to the connection.

In the project a POC package is developed for handling the network connection part of the application. The main components of this package are:

*MsgBase*: Indicates the format of messages used to communicate with the server. Mainly, it parses the messages that comes from the server and do the required operations according to these messages. Besides the objects from this class are used to send messages to the server. This class mainly handles protocol parsing.

*TCPConnection*: One instance from this class is created in order to handle the direct connection to the server. It just receives bytes from the server and passes this message to an instance of *MTCPConnectionObserver* object. In addition to this, it sends the bytes to the server that is passed by the instances of *MsgBase* class. The action of listening the messages send by the server is done in a separate thread in order not to suspend the application.

*MTCPConnectionObserver Interface*: Used for processing the messages transmitted by the *TCPConnection* object. It determines the type of the Message creates the proper *MsgBase* object. In the implementation a static instance of the *ChatApp* class implements this method.

ChatApp: Controls the messaging between the server and the client program. It also provides access to the user interface. It implements the *MTCPCConnectionObserver* interface.

ChatInfo: Holds the data information necessary for the application. This information can be stated as below:

*Me*: Information about the user. Such as Nick, telephone number, password, user id.

*ContactList*: Information about the users in the contact list.

*MessageList*: The messages left by his contacts to the user while he is offline.

When a user logs in to the system with his/her telephone number and password, a socket connection is established with the server and the server returns the *userId* to the client. As it is a socket connection, the *userId* does not have to be provided with every message send. Then the client wants the contact list and inbox information from the server. After gathering these information the *contactlist* page is shown to the user. Then the user may write to any of the users in his/her contact list, leave message to the ones that are offline, block/unblock the contacts, read the messages in the inbox, add or remove contacts. In order to add a specific contact to the contact list, the confirmation of this user is needed, so a confirmation message is send to these user asking if he/she accepts the request.

As a result of the project, a chat application is written which can run in different models of mobile phones. Besides, the user interface components developed can be used to write a variety of other programs. These components reduce the differences between the devices and make application development easier. The chat application enables the users with different devices to communicate over GPRS easily.

# İÇİNDEKİLER

MOBİL CHAT UYGULAMASI.....	1
MOBİL CHAT UYGULAMASI.....	3
MOBILE CHAT APPLICATION .....	4
İÇİNDEKİLER .....	9
1. GİRİŞ.....	11
2. PROJENİN TANIMI VE PLANI.....	13
2.1. Projenin amacı .....	13
2.2. Projenin kapsamı .....	13
2.3. Projeye ilişkin kestirimler.....	13
2.4. Zamanlama .....	14
2.5. Proje Kaynakları .....	14
3. KURAMSAL BİLGİLER.....	15
3.1. Java .....	15
3.2. Java 2 Micro Edition (J2ME) .....	18
3.2.1. Konfigürasyonlar ve profiller .....	18
3.2.1.1. Limitli Bağlı Aygıt Konfigürasyonu (CLDC-Connected Limited Device Configuration).....	20
3.2.1.2. Bağlı Aygıt Konfigürasyonu (CDC -Connected Device Configuration) ....	21
3.2.1.3. Mobil Aygıt Bilgilendirme Profili ( MIDP - Mobile Information Device Profile) .....	22
3.2.2. MIDP Programlama-Midletler.....	25
3.2.2.1. Midlet Yaşam Döngüsü .....	25
3.2.2.2. Midlet Geliştirme.....	26
3.2.2.3. Midletlerin Cep telefonuna Kurulumu.....	27
3.2.3. MIDP Kullanıcı Ara Birimi .....	28
3.3. J2ME Network Programlama .....	29
3.3.1. Generic Connection Framework (GCF) .....	29
3.3.2. MIDP Network Desteği .....	32
3.3.2.1. Socket .....	32
3.3.2.2. TCP/IP ve UDP/IP Haberleşme.....	33
3.3.2.3. Socket Connection Interface.....	34
4. ANALİZ VE MODELLEME .....	35
4.1. Kullanıcı Arayüz Bileşenleri .....	35
4.1.1. Arayüz nesneleri .....	35
4.1.2. Aksiyonlar.....	40
4.2. Network Bağlantı Sınıfları.....	44
4.2.1. MsgBase .....	44
4.2.2. TCPConnection.....	45
4.2.3. MTCPConnectionObserver Interface .....	45
4.2.4. ChatApp.....	45
4.2.5. ChatInfo .....	46
5. TASARIM, GERÇEKLEME VE TEST.....	48
5.1. Protokoller .....	48
5.1.1. Login.....	48
5.1.2. ContactList.....	48
5.1.3. Logout.....	48

5.2.	Test .....	49
5.3.	Sonuçlar .....	49
6.	SONUÇ ve ÖNERİLER .....	50
7.	KAYNAKLAR .....	51

# 1. GİRİŞ

Bu projede hedeflenen, deęişik özelliklere sahip cep telefonlarında çalışabilecek bir mesajlaşma uygulaması yazmaktır. Projenin amaçları aşağıdaki gibi verilebilir:

- o Deęişik ekran büyüklüklerine, bellek kısıtlamalarına ve tuş kombinasyonlarına sahip cihazların tümünde aynı şekilde görüntülenecek arayüz bileşenlerinin oluşturulması, arayüz bileşenlerinin yönetiminin sağlanması
- o Cep telefonundan sunucu bağlantısının gerçekleşmesi ve sunucu ile mesajlaşmanın sağlanması (Sunucu olarak MOBİNEX firmasına ait POC+ sunucusu kullanılmıştır)

Cep telefonlarının farklı ekran boyutları, tuş takımları ve bellek kısıtları, yazılan uygulamanın farklı cihazlarda bambaşka şekillerde görünmesi ve çalışmasına sebep olmaktadır. Bu durumu ortadan kaldırmaya yönelik olarak çeşitli uygulamalar geliştirilmiştir. Bunlara örnek olarak *Applications For Phones* ve *Java Polish* gösterilebilir. Fakat bu uygulamalar da her cep telefonu için ayrı bir *jar* dosyası oluşturma zorunluluğunu aşamamışlardır.

POC+ mesajlaşma programı, Symbian uyumlu cep telefonlarında kullanılabilir. Symbian OS işletim sistemine sahip cep telefonları için C++ programlama dili ile geliştirilmiş olan bu istemci uygulaması Symbian işletim sistemine sahip olmayan telefonlarda çalışmamaktadır. Yazılan uygulama bu kısıtlamayı ortadan kaldırmıştır ve Java MIDP 2.0 telefonların hepsinde çalışabilecek bir uygulama yazılmıştır.

Proje sonucunda, deęişik cep telefonu modellerinde çalışabilen bir chat uygulaması oluşturulmuştur. Ayrıca, oluşturulan arayüz elemanları deęişik türdeki bir çok uygulamanın yazılmasına olanak vermektedir. Oluşturulan bileşenler, cihazlar arasındaki farklılıkları azaltmakta ve uygulama geliştirmeyi kolaylaştırmaktadır. Oluşturulan chat uygulaması, farklı cihazlara sahip bireylerin uzak mesafelerden kolay haberleşmesini sağlamaktadır.