

JAVA API v2.0

Belge sürümü: 2.0.2

1. İindekiler

1.	İİNDEKİLER	2
2.	BU BELGENİN AMACI	3
3.	BELGE SÜRÜMLERİ	3
4.	SİSTEM GEREKSİNİMLERİ	3
5.	KULLANIM ŞEKLİ	4
5.1.	GENEL.....	4
5.2.	UYARILAR	4
5.3.	HATA KODLARI.....	5
5.4.	KULLANICI BİLGİLERİ KONTROLÜ	5
5.5.	MESAJ (SMS) GÖNDERİMİ	6
5.5.1.	<i>SMSTOMANY SINIFI</i>	7
5.5.2.	<i>SMSMULTISENDER SINIFI</i>	8
6.	BELGEDE GEEN MARKA VE TEKNOLOJİLER	10

2. Bu Belgenin Amacı

Bu belge, Toplu SMS gönderim hizmetimizi çift yönlü uygulamalar (mesaj toplama, oylama sistemleri, Müşteri İlişkileri Yönetimi vb.) geliştirmek için kullanmak isteyen yazılım geliştiricilerin, kullanmakta oldukları yazılım geliştirme ortamlarıyla Mobildev SMS Hizmetlerini entegre edebilmelerini sağlayan Interact API çözümümüzün nasıl kullanıldığını açıklamak için hazırlanmıştır.

3. Belge Sürümleri

Tarih	Sürüm	Açıklama
10.06.2003	1.0.0	İlk sürüm
18.08.2004	2.0.0	Yöntemlere ait açıklama bölümleri eklendi.
21.04.2005	2.0.1	Hata kodlarının açıklamaları düzeltildi.
14.02.2011	2.0.2	SetUsername yöntemine ait bilgi güncellemesi gerçekleşti.

4. Sistem Gereksinimleri

Mobildev JAVA API çözümünü kullanabilmeniz için gereken minimum sistem gereksinimleri aşağıda belirtilmiştir.

- Windows 98 SE, ME, 2000, XP, 2003, Linux, UNIX ya da MAC OS işletim sistemi
- Internet Explorer 6 SP1 (Windows Platformu İçin)
- JVM - JAVA Sanal Makinesi (Her işletim sistemi için)

5. Kullanım Şekli

5.1. Genel

Mobildev JAVA API, uygulama geliştiricilerin çözümlerini SMS teknolojisiyle sıfır maliyetli bir şekilde bütünleştirmelerini sağlamak ve platform bağımsız uygulamalar geliştirmelerine yardımcı olmak amacıyla geliştirilmiş bir JAVA çözümdür.

Yazılım geliştiriciler, Mobildev JAVA API'de sunulan yöntemleri, bu belgede belirtilen şekilde kullanarak çağırırlar. Çağrılan yöntemler <http://gateway.mobilus.net/com.mobilus> (Bundan sonra Gateway olarak anılacaktır.) adresine, taşımış oldukları parametreleri aktarırlar. Aktarılan parametreler Gateway tarafından işlenir. Eğer parametre bilgilerinde ya da gönderilen SMS paketinde hata oluşmuşsa, yakalanan hata HTTP üzerinden önceden tanımlı bir kod olarak göndericiye iletilir. Eğer bir hata oluşmamışsa gönderilmiş olan parametre ya da SMS paketine uygun dönüş ifadesi hazırlanıp HTTP üzerinden göndericiye iletilir. Önceden tanımlı hata kodları, belgenin ilgili ayrımında ayrıntılı olarak açıklanacaktır.

NOT: Mobildev JAVA API'yi kullanmak için, JAR paketini (Mobilus.jar) JAVA SDK'nın bulunduğu klasörün (Örn. C:\j2sdk1.4.2_04) altında bulunan "lib" klasörüne kopyalayın, CLASSPATH ortam değişkenine de JAR paketinin tam yolunu ekleyin. (Örn. C:\j2sdk1.4.2_04\lib\Mobilus.jar)

5.2. Uyarılar

Mobildev JAVA API çözümü aşağıdaki uyarılar göz önünde bulundurularak kullanılmalıdır.

- Mesaj içeriğinde Türkçe karakterler, LineFeed, Enter, Tab, "~", "€" gibi karakterler bulunmamalıdır.
- GSM numaraları 05326541220,5326541220 ya da 905326541220 şeklinde olmalıdır.

5.3. Hata Kodları

Önceden tanımlı hata kodları aşağıdaki tabloda ayrıntılarıyla verilmiştir.

Hata Kodu	Açıklama
01	Hatalı kullanıcı adı – şifre – bayi kodu
02	Yetersiz kredi (Mesaj gönderimi), Böyle bir Mesaj kodu (ID) yok, Paket işlenmemiş ya da Gateway tarafında beklemede (Raporlama)
03	Tanımsız Action değeri
04	Gelen XML yok
05	XML düğümü eksik ya da hatalı
06	Tanımsız Originator bilgisi
07	Mesaj kodu (ID) yok
08	Verilen tarihler arasında SMS gönderimi yok
09	Tarih alanları boş - hatalı
10	SMS gönderilemedi
11	Tanımlanamayan hata
13	Rapor istenen kullanıcı yok

Tablo 1 – Mobildev JAVA API Hata Kodları

5.4. Kullanıcı Bilgileri Kontrolü

Mobildev JAVA API, kullanıcı bilgileri kontrolü için CreditReporter adıyla bir sınıf sunar. Kullanıcı bilgilerini Gateway'den kontrol ettirmek için setUsername() yöntemine kullanıcı adınızı (username-company code), setPassword() yöntemine kullanıcı adınıza ait şifreyi parametre olarak girip SendMessage() yöntemini çağırdığınızda geri dönüş değeri olarak kredi ve originator bilgilerini alırsınız. Aşağıdaki örnekte CreditReporter sınıfının kullanımı gösterilmektedir.

Örnek 5.3.1 CreditReporter sınıfı

```
import com.Mobilus.Sms.*;
class merhaba
{
    public static void main(String[] args)
    {
        try
        {
            CreditReporter cr = new CreditReporter();
            cr.SetUsername("test-mb1000");
            cr.SetPassword("1111");
            Object[] o = cr.SendMessage();
            System.out.println((String)o[0]);
        }
        catch(Exception e)
        {
            e.printStackTrace();
        }
    }
}
```

CreditReporter sınıfını örnekte belirtildiği gibi Gateway'a gönderdiğinizde aşağıdaki geri dönüş değerlerini alırsınız:

1- İşlem başarılıysa:

Kontör sayısı<10>
Originator1<10>
Originator2<10>
OriginatorN

2- Hata oluşmuşsa: Bknz. Tablo 1

Geri dönüş değerlerinde bulunan <10> LineFeed karakterini temsil etmektedir. Originator bilgisi SMS gönderimlerinizde hedef GSM numarasının "Gönderen" kısmında çıkacak olan bilgiyi ifade etmektedir. Bu bilgi 11 karakterlik nümerik ya da alfanümerik karakterlerden oluşan (Türkçe ve &, <, > gibi özel karakterler içermeyen) bir ifade olabilir.

5.5. Mesaj (SMS) Gönderimi

Mobildev JAVA API, mesaj gönderimi için, iki adet sınıf sunmaktadır. Bunlar:

- 1- Aynı mesajı farklı numaralara göndermek için SmsToMany
- 2- Farklı mesajları farklı numaralara göndermek için SmsMultiSender

Mobildev olarak, JAVA API üzerinden göndereceğiniz toplu mesaj paketlerinin boyutunu 30.000'i aşmayacak şekilde ayarlamanızı öneririz.

5.5.1. SmsToMany Sınıfı

SmsToMany sınıfı, GSM numaralarına aynı içeriğe ait mesajın gönderilmesinde kullanılır. Bunu yapmak için SmsToMany sınıfından bir örnek aldıktan sonra:

- setUsername() yöntemine kullanıcı adını,
- setPassword() yöntemine kullanıcı adına ait şifreyi,
- setOriginator() yöntemine Originator bilgisini,
- setMessage() yöntemine en fazla 160 karakter uzunluğundaki (Türkçe ve "&", "<", ">", ENTER, LineFeed gibi özel karakterler içermeyen) mesaj içeriğini,
- Eğer yapacağınız gönderimin ileri bir tarihte gerçekleşmesini istiyorsanız setDate() yöntemine
- GünAyYılSaatDakika (ggaayyyssdd) şeklinde tarih bilgisini,
- setNumbers() yöntemine mesajın gönderileceği GSM numaralarını belirttikten sonra SendMessage() yöntemini çağırırsanız mesaj paketi Gateway'a gönderilir. Gelen paket Gateway tarafından işlendikten sonra uygun geri dönüş değeri oluşturulur. Geri dönüş değerini SendMessage() yöntemini bir değışkene aktararak kullanabilirsiniz.

Aşağıdaki örnekte SmsToMany sınıfının kullanımıyla ilgili örnek kod görölmektedir.

Örnek 5.5.1.1 SmsToMany sınıfı

```
import com.Mobilus.Sms.*;
class merhaba
{
    public static void main(String[] args)
    {
        try
        {
            SmsToMany cr = new SmsToMany();
            cr.setUsername("test-mb1000");
            cr.setPassword("1111");
            cr.setOriginator("SMSTEST");
            cr.setMessage("Test mesajıdır");
            cr.setDate("181120051455");
            String[] numbers = {"05334924505,05556446020"};
            cr.setNumbers(numbers);
            Object[] o = cr.SendMessage();
            System.out.println((String)o[0]);
        }
        catch(Exception e)
        {
            e.printStackTrace();
        }
    }
}
```

SendMessage() yöntemiyle gönderilen SMS paketi Gateway'a ulaştığında iki çeşit geri dönüş değeri alınır:

- 1- İşlem başarılıysa: "ID: 3152005" şeklinde SMS paketine ait TimerID bilgisi döner.
- 2- Hata oluşmuşsa: Bknz. Tablo 1

5.5.2. SMSMultiSender Sınıfı

SmsMultiSender sınıfı, GSM numaralarına farklı içerikli mesajlar gönderilmesinde kullanılır. Bunu yapmak için SmsMultiSender sınıfından bir örnek aldıktan sonra:

- SetUsername() yöntemine kullanıcı adını,
- SetPassword() yöntemine kullanıcı adına ait şifreyi,
- SetOriginator() yöntemine Originator bilgisini,
- AddMessage() yöntemine en fazla 160 karakter uzunluğundaki (Türkçe ve "&", "<", ">", ENTER, LineFeed gibi özel karakterler içermeyen) mesaj içeriğini ve mesajın gönderileceği cep telefonu numarasını,
- Eğer yapacağınız gönderimin ileri bir tarihte gerçekleşmesini istiyorsanız SetDate() yöntemine GünAyYılSaatDakika (ggaayyyssdd) şeklinde tarih bilgisini belirttikten sonra SendMessage() yöntemini çağırırsanız mesaj paketi Gateway'a gönderilir. Gelen paket Gateway tarafından işlendikten sonra uygun geri dönüş değeri oluşturulur. Geri dönüş değerini SendMessage() yöntemini bir değişkene aktararak kullanabilirsiniz.

Aşağıdaki örnekte SmsMultiSender sınıfının kullanımıyla ilgili örnek kod görülmektedir.

Örnek 5.5.2.1 SmsMultiSender sınıfı

```
import com.Mobilus.Sms.*;
class merhaba
{
    public static void main(String[] args)
    {
        try
        {
            SmsToMany cr = new SmsToMany();
            cr.SetUsername("test-mb1000");
            cr.SetPassword("1111");
            cr.SetOriginator("SMSTEST");
            cr.SetDate("181120051455");
            cr.AddMessage("05356446022","Test mesajidir bu.");
            cr.AddMessage("05428112345","multi test mesajı");
            Object[] o = cr.SendMessage();
            System.out.println((String)o[0]);
        }
        catch(Exception e)
        {
            e.printStackTrace();
        }
    }
}
```


SendMessage() yöntemiyle gönderilen SMS paketi Gateway'a ulaştığında iki çeşit geri dönüş değeri alınır:

- 1- İşlem başarılıysa: "ID: 3152005" şeklinde SMS paketine ait TimerID bilgisi döner.
- 2- Hata oluşmuşsa: Bknz. Tablo 1

5.6. Raporlama

Mobildev JAVA API, mesaj gönderimlerinin raporlanması için SmsReporter adıyla bir sınıf sunar. Mesaj gönderiminden sonra geri dönüş değeri olarak aldığınız mesaj gönderim kodunu SmsReporter sınıfının SetMsgid() yöntemine parametre olarak eklerseniz geri dönüş olarak gönderim raporunu alırsınız.

Aşağıdaki örnekte SmsReporter sınıfı gösterilmektedir.

Örnek 5.5.1 SmsReporter sınıfı

```
import com.Mobilus.Sms.*;
class merhaba
{
    public static void main(String[] args)
    {
        try
        {
            SmsReporter cr = new SmsReporter();
            cr.SetUsername("test-mb1000");
            cr.SetPassword("1111");
            cr.SetMsgid("3152028");
            Object[] o = cr.SendMessage();
            System.out.println((String)o[0]);
        }
        catch(Exception e)
        {
            e.printStackTrace();
        }
    }
}
```

Bilgiler Gateway'a ulaştığında iki çeşit geri dönüş değeri alınır:

- 1- İşlem başarılıysa:
"TimerID<32>GSM Numarası<32>Durum<10>" şeklinde (Bknz. Örnek 5.5.2)

Örnek 5.5.2 Gönderilen mesaj bazında (TimerID) raporlama geri dönüş değeri

```
3152028<32>905448838799<32>2<10>
3152028<32>905378838799<32>3<10>
3152028<32>905058838799<32>1<10>
```

NOT: Durum bilgisinde: “1” mesajın beklemede olduğunu, “2” gönderildiğini, “3” ise iletilmediğini belirtir. Geri dönüş değerlerinin tümünde bulunan <32> boşluk karakterini, <10> LineFeed karakterini temsil etmektedir.

2- Hata oluşmuşsa: Bknz. Tablo 1

6. Belgede Geçen Marka ve Teknolojiler

JAVA	Sun Micro Systems’in ticari markası
JAVA (JAVA Sanal Makinesi)	Sun Micro Systems’in ticari markası
HTTP	Hypertext Transfer Protocol
Windows®	Microsoft Corporation tescilli markası
Internet Explorer (Sadece logo)	Microsoft Corporation tescilli markası