

■ connecting your business



Integration Google Calendar

LANCOM Wireless ePaper Server

Inhalt

1 Zielsetzung und Voraussetzungen.....	3
1.1 Zielsetzung.....	3
1.2 Voraussetzungen.....	3
2 Google Developers Console & Google Calendar.....	5
2.1 Projekt erstellen.....	5
2.2 APIs definieren.....	5
2.3 Einrichtung eines Dienstkontos für den Kalenderzugriff.....	6
2.4 Konfiguration von Google Calendar.....	7
3 Java-Programm WePService.....	8
3.1 Installation.....	8
3.2 Inhalt des Ordners WePService.....	8
3.3 WePService.jar.....	8
3.4 calendar.properties.....	8
3.5 client.properties.....	9
3.6 service.properties.....	10
3.7 run.bat.....	11
3.8 run.sh.....	11
4 Template für Raumbeschilderung.....	12
4.1 Beispiel eines Templates.....	12
4.2 Erläuterungen einzelner Codeabschnitte.....	13
4.3 Interaktion von Template, XML-Informationen und Wireless ePaper Displays.....	15
5 Quell-Code des Java-Programms WePService.....	17

1 Zielsetzung und Voraussetzungen

1.1 Zielsetzung

Ziel dieses Dokumentes ist es, Ihnen eine Möglichkeit aufzuzeigen, wie Sie erfolgreich die LANCOM Wireless ePaper Solutions in ein Umfeld integrieren, in dem Google Calendar genutzt wird. Dies erfolgt anhand eines Java-Programms, den zugehörigen Konfigurationsdateien und einem XSL-Template.

Bei dem Java-Programm, den Konfigurationsdateien und dem Template handelt es sich um Beispiele, welche die Erstellung eigener Dateien vereinfachen soll. Soll der Code produktiv eingesetzt werden, ist es erforderlich, Angaben in der Konfigurationsdatei entsprechend abzuändern. Weitere Templates können erstellt, beziehungsweise genutzt werden, um eine optimale Anpassung für den geplanten Einsatz zu erreichen. Des Weiteren kann das Java-Programm neu- oder umgeschrieben werden, wenn das bereits vorhandene nicht ausreichen sollte.

1.2 Voraussetzungen

Die erfolgreiche Verknüpfung der LANCOM Wireless ePaper Solutions mit einem Google Calendar erfordert gewisse Voraussetzungen an vorhandener Software und deren Anpassung.

1. LANCOM Wireless ePaper Server

Folgende Voraussetzungen müssen bei dem LANCOM Wireless ePaper Server gegeben sein:

- Das zu nutzende Template muss unter `\\<Installationsverzeichnis>\data\templates\` abgelegt werden.
- Bilder, auf die im Template verwiesen wird, müssen unter `\\<Installationsverzeichnis>\data\images\` abgelegt werden.
- Jedem Wireless ePaper Display muss genau ein Tag zugewiesen werden, welches den Raumnamen enthält. Es dürfen nur Großbuchstaben verwendet werden. Etwaige Leerzeichen in der Raumbezeichnung (Kalendernamen) müssen durch Unterstriche ersetzt werden. Beispiel: Raumbezeichnung in Google Calendar Raum AACHEN resultiert in Tag RAUM_AACHEN.

2. Google Developers Console & Google Calendar

Folgende Voraussetzungen müssen in der Google Developers Console und dem Google Calendar des zu nutzenden Google Accounts geschaffen werden:

- In der Google Developers Console muss ein Projekt für die Wireless ePaper Verwaltung angelegt werden (siehe [Kapitel 2.1](#)), in dem die notwendige API aktiviert ist (siehe [Kapitel 2.2](#)).
- In der Google Developers Console muss in dem zuvor erstellten Projekt für die Wireless ePaper Verwaltung ein Dienstkonto angelegt werden, mit dessen Zugangsdaten das Java-Programm auf den Google Calendar zugreift (siehe [Kapitel 2.3](#)).
- In Google Calendar muss für jeden Kalender das eingerichtete Dienstkonto Lesezugriff erhalten (siehe [Kapitel 2.4](#)).

3. Java-Programm: WePService

Folgende Voraussetzungen müssen für die Nutzung des Java-Programms WePService gegeben sein:

- Es muss ein System mit Java JDK (Version 8 oder höher) im Netzwerk vorhanden sein, auf dem das Java-Programm ausgeführt werden kann.

1 Zielsetzung und Voraussetzungen



openJDK wird nicht unterstützt.

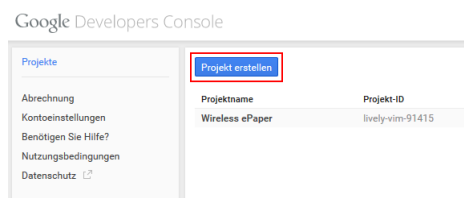
- Die Konfigurationsdateien müssen auf die Anforderungen der geplanten Nutzung angepasst werden.

2 Google Developers Console & Google Calendar

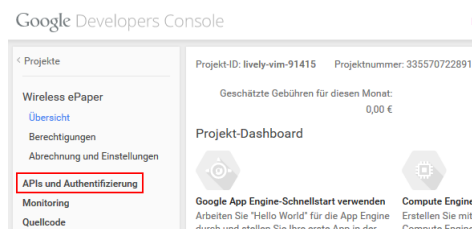
Zu Beginn muss in der Google Developers Console ein Projekt angelegt werden, in dem die zu nutzenden APIs und Zugangsdaten konfiguriert werden. Hierzu rufen Sie in Ihrem Browser die Webseite <https://console.developers.google.com> auf. Melden Sie sich mit den Zugangsdaten des Google Accounts zur Kalenderverwaltung an.

2.1 Projekt erstellen

Mit einem Klick auf die Schaltfläche **Projekt erstellen** wird ein neues Projekt erstellt. Die Projekt-Bezeichnung können Sie frei wählen.

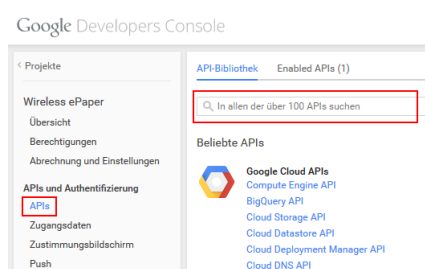


Klicken Sie den Projektnamen in der Übersicht an, um in das Projekt-Dashboard zu wechseln. Unter dem Menüpunkt **APIs und Authentifizierung** finden Sie die Konfigurationspunkte, mit denen die zu nutzenden APIs definiert und Dienstkonten erstellt werden können.

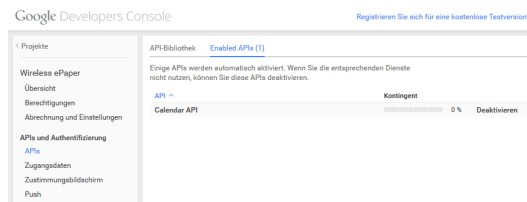


2.2 APIs definieren

Im Unterpunkt **APIs > API-Bibliothek** geben Sie **Calendar API** als Suchbegriff ein und aktivieren die API mit gleichem Namen.

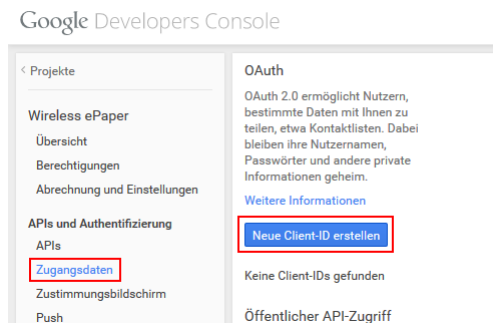


Im Unterpunkt **APIs > Enabled APIs** deaktivieren Sie alle APIs bis auf die **Calendar API**.



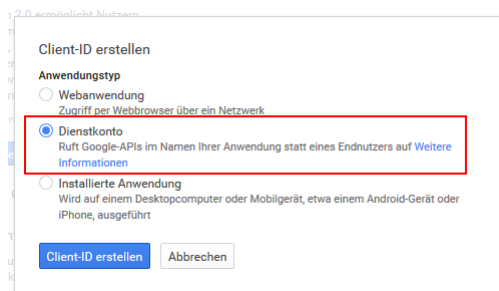
2.3 Einrichtung eines Dienstkontos für den Kalenderzugriff

Im Unterpunkt **Zugangsdaten** sehen Sie bereits erstellte Konten und können neue anlegen. Klicken Sie auf die Schaltfläche **Neue Client-ID erstellen**, um ein neues Konto anzulegen.



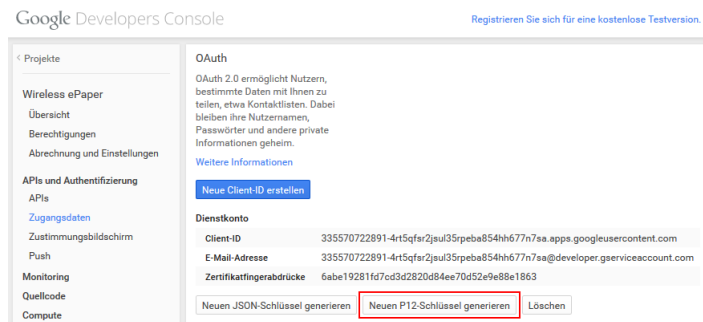
Wählen Sie die Option **Dienstkonto** aus und bestätigen Sie die Auswahl mit einem Klick auf die Schaltfläche **Client-ID erstellen**. Hiermit wird automatisch der Download eines Schlüssels im JSON-Format gestartet.

! Dieser Schlüssel wird nicht benötigt. Den Download können Sie daher abbrechen.



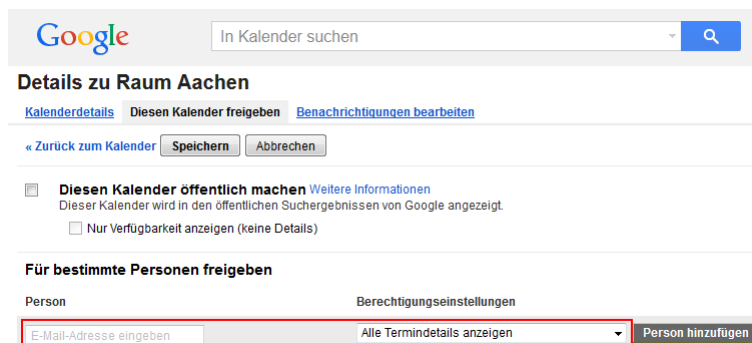
Das Dienstkonto besteht aus Client-ID, E-Mail-Adresse und Zertifikatsfingerabdrücken. Erstellen Sie eine neue PKCS12-Datei, indem Sie auf die Schaltfläche **Neuen P12-Schlüssel generieren** klicken. Dies startet den Download-Prozess der PKCS12-Datei, welche Sie speichern. Sie wird von dem Java-Programm WePService genutzt, um sich am Google Calendar

zu authentifizieren. Löschen Sie danach den Zertifikatsfingerabdruck, der vor der Erstellung vorhanden war. Dieser ist mit dem automatisch generierten Schlüssel im JSON-Format verknüpft und wird nicht benötigt.



2.4 Konfiguration von Google Calendar

Es ist erforderlich, das erstellte Dienstkonto mit den einzelnen Kalendern der Räume in Google Calendar zu verknüpfen. Hierzu öffnen Sie unter <https://www.google.com/calendar> Google Calendar in Ihrem Browser. Melden Sie sich mit den Zugangsdaten des Google Accounts zur Kalenderverwaltung an. Tragen Sie in den jeweiligen Kalendern unter **Kalender-Einstellungen > Diesen Kalender freigeben** die E-Mail-Adresse des Dienstkontos ein. Setzen Sie die Berechtigungseinstellung auf **Alle Termindetails anzeigen**.



Die E-Mail-Adresse des Dienstkontos finden Sie in der Google Developers Console im entsprechenden Projekt unter **APIs und Authentifizierung > Zugangsdaten > OAuth > Dienstkonto**.

3 Java-Programm WePService

3.1 Installation

Es genügt ein einfaches Ablegen des Ordners "WePService" auf dem System, auf dem es ausgeführt werden soll. Die PKCS12-Datei, die bei der Erstellung des Dienstkontos (siehe [Kapitel 2.3](#)) heruntergeladen wurde, muss in diesen Ordner kopiert und in `calendar.p12` umbenannt werden.

3.2 Inhalt des Ordners WePService

Datei	Bedeutung
WePService.jar	Das Java-Programm zur Abfrage von Kalendereinträgen und Konfigurationsdatei für Kalendereigenschaften.
calendar.properties	Konfigurationsdatei für die Parameter von Google Calendar.
client.properties	Konfigurationsdatei für die Parameter des LANCOM Wireless ePaper Servers.
service.properties	Konfigurationsdatei für das Template und das Update-Intervall.
run.bat	Batch-Datei zum Start des Java-Programms unter Windows
run.sh	Shell-Datei zum Start des Java-Programms unter Linux
log4j2.xml	XML-Datei als Basis zur Log-Erstellung
calendar.p12	Schlüssel für das Dienstkonto mit Zugriff auf Google Calendar. Der Schlüssel wird in Kapitel 2.3 erstellt und ist nicht Teil des Auslieferungszustandes.

3.3 WePService.jar

Hierbei handelt es sich um das Java-Programm, welches auf Google Calendar zugreift. Das Programm fragt die Informationen der einzelnen Kalender ab und schickt notwendige Aktualisierungen an den LANCOM Wireless ePaper Server. Auf eine detaillierte Erläuterung des Codes wird an dieser Stelle verzichtet, da alle notwendigen Änderungen für die Nutzung über die Konfigurationsdateien erfolgen.

3.4 calendar.properties

Die Datei `calendar.properties` ist eine Konfigurationsdatei, in welcher die notwendigen Parameter betreffs Google Calendar eingetragen sind.

```
# the used timezone when computing dates
time_zone=Europe/Berlin
# the number of events to be read from each calendar
number_events=2
```



```
# the google service account id used for fetching calendar data
serviceaccountid=

# list of resources (rooms) that will be read

# Room 0
resource0=
# Room 1
resource1=
# Room 2
resource2=
# Room 3
resource3=
# Room 4
resource4=
```

Parameterbeschreibung:

Parameter	Beschreibung
time_zone	Bestimmt die Zeitzone. Default: Europe/Berlin
number_events	Bestimmt die Anzahl der abgefragten Termine pro Kalender. Default: 2
serviceaccountid	Enthält die E-Mail-Adresse des Dienstkontos, welches in Kapitel 2.3 erstellt wurde und zur Abfrage der Kalenderinformationen genutzt wird. Den zugehörigen P12-Schlüssel (PKCS12-Datei) legen Sie bitte im Verzeichnis "WePService" ab und benennen die Datei in <code>calendar.p12</code> um.
ressource[x]	Enthält Kalenderadresse für Raum x, dessen Informationen abgefragt werden sollen. Die Raum-Nummerierung beginnt bei 0. Die Kalenderadresse ist in Google Calendar unter Kalender-Einstellungen > Kalenderdetails > Kalenderadresse einsehbar.
run.bat	Batch-Datei zum Start des Java-Programms unter Windows.
run.sh	Shell-Datei zum Start des Java-Programms unter Linux.
log4j2.xml	XML-Datei als Basis zur Log-Erstellung.
calendar.p12	Schlüssel für das Dienstkonto mit Zugriff auf Google Calendar. Der Schlüssel wird in Kapitel 2.3 erstellt und ist nicht Teil des Auslieferungszustands.

3.5 client.properties

Die Datei `client.properties` ist eine Konfigurationsdatei, in welcher die notwendigen Parameter zur Kommunikation mit dem LANCOM Wireless ePaper Server hinterlegt sind.

```
# the WeP-Server's hostname or ip address
host = 192.168.1.1
# the WeP-Server's port
port = 8001
# the WeP-Server's username
user = admin
# the WeP-Server's password
password = admin
# the annotation the client send its data (XML or JSON - XML is recommended)
annotation = XML
```

Parameterbeschreibung:

Parameter	Beschreibung
host	Enthält die IPv4-Adresse des LANCOM Wireless ePaper Servers.
port	Enthält den Port, über den mit dem LANCOM Wireless ePaper Servers kommuniziert wird. Default: 8001 (Es wird empfohlen, die Voreinstellung beizubehalten.)
user	Enthält den Benutzernamen für die Kommunikation mit dem LANCOM Wireless ePaper Server. Default: admin (Es wird empfohlen, die Voreinstellung beizubehalten.)
password	Enthält das Passwort zum oben angegebenen user. Default: admin (Es wird empfohlen, die Voreinstellung beizubehalten.)
annotation	Enthält das Format der Kommunikation mit dem LANCOM Wireless ePaper Server. Default: XML (Es wird empfohlen, die Voreinstellung beizubehalten.)

3.6 service.properties

Die Datei `service.properties` ist eine Konfigurationsdatei, in welcher weitere Parameter definiert werden.

```
# defines the xsl template file on the WeP server used to generate the image on the WeP
template = lcsconference_landscape.xsl

# display text aliases for today and tomorrow instead of numeric dates
use.date.alias = false

# the update interval of the server in milliseconds
# 30 sec.
#update.interval = 30000
# 1 min
#update.interval = 60000
# 5 mins
update.interval = 300000
```

Parameterbeschreibung:

Parameter	Beschreibung
template	Bestimmt das Template, welches der Wireless ePaper Server zur Erstellung der Bilder für die Wireless ePaper Displays nutzt. Es ist erforderlich, das Template auf dem Wireless ePaper Server unter <code>\\<Installationsverzeichnis>\data\templates\</code> abzulegen. Default: <code>lcsconference_landscape.xsl</code> (Es wird empfohlen, die Voreinstellung beizubehalten.)
use.date.alias	Enthält die Einstellung, ob anstelle des numerischen Datums ein Text-Alias (Today, Tomorrow) genutzt werden soll. Default: <code>false</code> (Es wird das numerische Datum angezeigt.)
update.interval	Bestimmt das Intervall in Millisekunden, in dem das Java-Programm die Informationen bei Google Calendar abfragt.

3.7 run.bat

Die Datei `run.bat` ist eine Batch-Datei, welche zum Start des Java-Programms "WePService" unter Windows genutzt wird. Das Programm fragt die Kalenderinformationen von Google Calendar ab (siehe [Kapitel 3.3](#)).

3.8 run.sh

Die Datei `run.sh` ist eine Shell-Datei, welche zum Start des Java-Programms "WePService" unter Linux genutzt wird. Das Programm fragt die Kalenderinformationen von Google Calendar ab (siehe [Kapitel 3.3](#)).



Beachten Sie, dass die Datei im Auslieferungszustand noch nicht ausführbar ist.

4 Template für Raumbeschilderung

Das Template ist als XSL-Datei auf dem Wireless ePaper Server abgelegt. Es gibt das Format vor, wie die von dem Java-Programm "WePService" übermittelten Updates vom Wireless ePaper Server gerendert werden. Das resultierende Bild wird dann an die entsprechenden Wireless ePaper Displays übermittelt.

4.1 Beispiel eines Templates

Dieses XSL-Template zeigt eine mögliche Beschilderung von Meetingräumen am Beispiel der LANCOM Systems GmbH.

! Die Zeilennummerierung ist nicht Bestandteil des Codes und dient lediglich zur Verbesserung der Übersichtlichkeit.

```

001 <?xml version="1.0" encoding="UTF-8"?>
002 <!--Das Template ist für 7,4" Displays vorgesehen, die in horizontaler Ausrichtung
angebracht sind.-->
003 <xsl:stylesheet xmlns:xsl="http://www.w3.org/1999/XSL/Transform" version="1.0">
004
005 <xsl:template match="Record">
006
007 <!-- Render-Informationen für 7,4" Displays -->
008 <image height="480" width="800" rotation="90" font-family="Verdana">
009
010 <!-- Room -->
011 <field height="108" width="780" x="10" y="20">
012 <text align="center" font-size="40" font-weight="bold">
013 <utils method="toUpperCase">
014 <xsl:value-of select="room/@roomName"/>
015 </utils>
016 </text>
017
018 <!-- Date -->
019 <text align="center" font-size="35" font-weight="bold" padding-top="10">
020 <xsl:value-of select="room/field[@key='date']/@value"/>
021 </text>
022 </field>
023
024 <line thickness="2" x-from="0" x-to="800" y-from="130" y-to="130"/>
025
026 <!-- Time1 -->
027 <field height="50" width="780" x="20" y="150">
028 <text align="left" font-weight="bold" font-size="40">
029 <xsl:value-of select="room/field[@key='time1']/@value"/>
030 </text>
031 </field>
032
033 <!-- Purposel -->
034 <field height="50" width="780" x="20" y="200">
035 <text align="left" font-size="40" condense="1, 0.8, 0.6, 0.5">
036 <xsl:value-of select="room/field[@key='purposel']/@value"/>
037 </text>
038 </field>
039
040 <!-- Chair1 -->
041 <field height="40" width="770" x="20" y="250">
042 <text align="left" font-weight="bold" font-size="30">

```

```

043     <xsl:value-of select="room/field[@key='chair1']/@value"/>
044   </text>
045 </field>
046
047   <!-- Time2 -->
048   <field height="35" width="780" x="20" y="320">
049     <text align="left" font-size="28">
050       <xsl:value-of select="room/field[@key='time2']/@value"/>
051     </text>
052   </field>
053
054   <!-- Purpose2 -->
055   <field height="35" width="780" x="20" y="355">
056     <text align="left" font-size="28" condense="1, 0.8, 0.6, 0.5">
057       <xsl:value-of select="room/field[@key='purpose2']/@value"/>
058     </text>
059   </field>
060
061   <!-- Chair2 -->
062   <field height="30" width="770" x="20" y="390">
063     <text align="left" font-size="20">
064       <xsl:value-of select="room/field[@key='chair2']/@value"/>
065     </text>
066   </field>
067
068   <!-- LANCOM Logo -->
069   <field align="right" height="60" width="780" x="10" y="410">
070     </img>
071   </field>
072 </image>
073 </xsl:template>
074 </xsl:stylesheet>

```

4.2 Erläuterungen einzelner Codeabschnitte

In diesem Kapitel werden einzelne Code-Abschnitte des gezeigten XSL-Templates erläutert.

Display-Informationen

```

007   <!-- Render-Informationen für 7,4" Displays -->
008   <image height="480" width="800" rotation="90" font-family="Verdana">

```

Hier wird definiert, für welches Wireless ePaper Display der darauf folgende Code-Abschnitt gültig ist. In diesem Fall handelt es sich um ein 7,4"-Display mit einer Auflösung von 800 x 480 Pixeln.

<code>height</code>	Bestimmt die Höhe des Bildes in Pixeln.
<code>width</code>	Bestimmt die Breite des Bildes in Pixeln.
<code>rotation</code>	Bestimmt die Rotation des Bildes in Grad. 0 entspricht der vertikalen Montierung und 90 der horizontalen Montierung.
<code>font-family</code>	Bestimmt die genutzte Schriftart.

Definition von Textfeldern

```

010   <!-- Room -->
011   <field height="108" width="780" x="10" y="20">
012     <text align="center" font-size="40" font-weight="bold">
013       <utils method="toUpperCase">
014         <xsl:value-of select="room/@roomName"/>

```

```

015     </utils>
016   </text>
017
018   <!-- Date -->
019   <text align="center" font-size="35" font-weight="bold" padding-top="10">
020     <xsl:value-of select="room/field[@key='date']/@value"/>
021   </text>
022 </field>

```

In diesem Bereich wird die Positionierung der Raumbezeichnung und des Datums definiert.

Hierzu wird zuerst ein Feld mit Höhe, Breite und Positionierung auf dem Display definiert:

```
<field height="108" width="780" x="10" y="20">.
```

<code>height</code>	Bestimmt die Höhe des Feldes in Pixeln.
<code>width</code>	Bestimmt die Breite des Feldes in Pixeln.
<code>x</code>	Bestimmt den Abstand zum linken Rand des Displays in Pixeln.
<code>y</code>	Bestimmt den Abstand zum oberen Rand des Displays in Pixeln.

Innerhalb dieses Feldes wird das Layout der verschiedenen Texte definiert:

```
<text align="center" font-size="40" font-weight="bold">
```

<code>align</code>	Bestimmt, wie der Text angeordnet ist. (z.B. center, left, right).
<code>font-size</code>	Bestimmt die Schriftgröße.
<code>font-weight</code>	Bestimmt den Schriftschnitt (z.B. bold, italic).
<code>y</code>	Bestimmt den Abstand zum oberen Rand des Displays in Pixeln.

```
<utils method="toUpperCase">
```

Der Aufruf dieser Methode erzwingt die Darstellung des Textes in Großbuchstaben.

```
<xsl:value-of select="room/@roomName"/>
```

Hier wird auf die von der Skriptbibliothek übermittelten XML-Informationen zugegriffen und der Text an der entsprechenden Stelle eingesetzt. Die bisherigen Definitionen haben nur das Layout vorgegeben.

Daraufhin wird nach gleichem Prinzip der Text definiert, in dem das übermittelte Datum angezeigt wird. Weitere Felder enthalten das Layout für die restlichen übermittelten Informationen für die einzelnen Meetings.

Einbindung von Grafiken

```

068   <!-- LANCOM Logo -->
069   <field align="right" height="60" width="780" x="10" y="410">
070     </img>
071   </field>

```

Zusätzlich zu Textfeldern können auch Grafiken eingebunden werden. Hier wird wie bereits oben zunächst ein Feld definiert, um die Position der Grafik festzulegen.

```
</img>
```

Zusätzlich wird nun hiermit der Speicherort der Grafik und deren Dateiname angegeben.

4.3 Interaktion von Template, XML-Informationen und Wireless ePaper Displays

Die Interaktion von Template, XML-Informationen und der finalen Darstellung der Wireless ePaper Displays wird in diesem Kapitel exemplarisch gezeigt.

Schritt 1:

Wird durch das Java-Programm "WePService" ein Update ausgelöst, enthält der Wireless ePaper Server XML-Informationen mit folgendem Inhalt:

! Die Zeilennummerierung ist nicht Bestandteil des Codes und dient lediglich zur Verbesserung der Übersichtlichkeit.

```
001 <?xml version="1.0" encoding="UTF-8" standalone="yes"?>
002 <TaskOrder title="Refresh D1001BF6 for Aachen B2.100">
003 <TemplateTask labelId="D1001BF6" externalId="4711"
004   template="lcsconference_landscape.xsl">
005   <room roomName="Aachen B2.100">
006     <field key="date" value="16.09.2014"/>
007     <field key="time1" value="10:00 - 11:30"/>
008     <field key="purpose1" value="Projektkoordination Marketing"/>
009     <field key="chair1" value="Helmut Müller"/>
010     <field key="time2" value="11:30 - 13.00"/>
011     <field key="purpose2" value="Team-Besprechung Controlling"/>
012     <field key="chair2" value="Karin Klein"/>
013   </room>
014 </TemplateTask>
015 </TaskOrder>
```

Schritt 2:

Der Wireless ePaper Server wertet die erhaltenen Informationen Zeile für Zeile aus. So werden das betroffene Display und das benötigte Template bestimmt. Anhand des Templates erstellt der Server nun den Datensatz, aus dem das spätere Bild gerendert wird. Das Layout wird durch das Template bestimmt und um die Informationen aus dem XML ergänzt.

Zu Illustrationszwecken werden zwei Inhaltselemente erläutert: die Leitung des Folgemeetings und das Firmenlogo.

Die Leitung des Folgemeetings wird laut Template wie folgt formatiert:

```
061 <!-- Chair2 -->
062 <field height="30" width="770" x="20" y="390">
063   <text align="left" font-size="20">
064     <xsl:value-of select="room/field[@key='chair2']/@value"/>
065   </text>
066 </field>
```

Die notwendige Information für die Darstellung findet sich im XML unter:

```
011 <field key="chair2" value="Karin Klein"/>
```

Für die im Template hinterlegte Grafik muss nicht auf die XML-Informationen zugegriffen werden. Stattdessen wird die Pfadangabe ausgewertet.

```
068 <!-- LANCOM Logo -->
069 <field align="right" height="60" width="780" x="10" y="410">
070   </img>
071 </field>
```

Schritt 3:

Rendern des Bildes und anschließendes Senden an das Wireless ePaper Display. Das Resultat ist die folgende Darstellung, farblich hervorgehoben sind die Leitung des Folgemeetings (1) und die Grafik (2).



5 Quell-Code des Java-Programms WePService

Den Quell-Code für das Java-Programm "WePService" finden Sie in der ZIP-Datei im Unterordner "SourceCode".