WapIDE SDK
File
Help
WapIDE
Wireless Application Protocol
Integrated Development Environment
Click on an icon to run a tool.
Ready.
Browser
App Designer
Server Toolset
Service Development Kit
© 1998-2000 Ericsson Radio Systems AB

WapIDE User Guide

ERICSSON

http://www.ericsson.com/developerszone
Version March 22, 2000

Legal Notice

Ericsson is the leading provider in the new telecom world, with communication solutions that combine telecom and datacom technologies with freedom of mobility for the user. With more than 100,000 employees in 140 countries, Ericsson simplifies communications for its customer – network operator, service providers, enterprises and consumers – the world over.

We welcome comments. Please send them to:

Support at Ericsson Developers' Zone at http://www.ericsson.com/developerszone

This document is part of the Ericsson WapIDE Service Development Kit.

The content of this document may be changed without prior notice.

Contents

Introduction

This document provides information how to install and use the Ericsson WAP Software Development Kit (WapIDE) to be able to create services for a WAP platform.

WapIDE facilitates the creation of applications using WAP, it enables operators, application developers, or any interested party to develop and test real WAP applications swiftly and easily. WapIDE can be downloaded free of charge from Ericsson Developers' Zone, which is found at http://www.ericsson.com/developerszone.

This document assumes that the user has a basic knowledge of WAP. Otherwise we refer to the WAP resources listed in "Related documents and other resources".

This document is a companion to the help files supplied with WapIDE installation. See those for further details.

This document relates to WapIDE version: WapIDE 2.1

Ericsson WapIDE

WapIDE is an SDK (Service Development Kit). It is a complete development environment that enables third party companies to develop, and test WAP applications quickly and easily. WapIDE consists of three different tools:

- *The Application Designer* – allows developers to create, compile and test WML and WMLScript applications quickly and easily.
- *The Browser* – allows the user to access WML decks and cards using a simulated WAP device. It includes a graphical UI, WML and WMLScript bytecode interpreters, and simulated WTA event handling.
- *The Server Toolset* – these tools are the compilers and libraries (included with the software) which let you try your dynamic WAP applications and helps you with sample code examples.

Related documents and other resources

Related Internet sites

Ericsson	http://www.ericsson.com
Ericsson Developers' Zone	http://www.ericsson.com/developerszone
Mobile Internet	http://www.mobileinternet.ericsson.com
WAP Forum	http://www.wapforum.org/

Erlang Systems	http://www.erlang.se
Xitami	http://www.imatix.com
TCL/TK	http://www.scripties.com

Related documents

Mobile Phone R380 Design Guidelines for WAP Services
Found at http://www.ericsson.com/developerszone

Mobile Phone R320 Design Guidelines for WAP Services
Found at http://www.ericsson.com/developerszone

Mobile Phone MC218 Design Guidelines for WAP Services
Found at http://www.ericsson.com/developerszone

Support

WapIDE comes with a help file that can be accessed directly from the Start menu in the taskbar (Start / Programs / Ericsson / WapIDE) or from inside WapIDE by using the help file menu.

Support can be found at the Ericsson Developers' Zone (see "Related Internet sites") free of charge.

Typographical conventions

The following typographical conventions are used in this document:

Bold	Names of commands in menus, buttons.
Italic	Specific terminology
`Courier`	Computer text, file names

Installation

This chapter describes the system requirements and installation procedure of WapIDE.

The WapIDE can be found at Ericsson Developers' Zone (*WAP -> Developer Tools*), at http://www.ericsson.com/developerszone.

To access the zone you have to register at the web site. The registration gives you access to the documentation and developer resources located at the site.

System Requirements

Operating Systems

WapIDE runs on the following:

- Windows NT version 4 or Windows 95/98

Requirements:

- > 10 MB HD
- 32 MB RAM

The software products included are:

- WapIDE
- Perl5
- Xitami Web Server
- TCL/TK

Installation procedure

WapIDE consists of two parts:

- WapIDE_3PP_2_0.exe
- WapIDE_SDK_2_1.exe

It is important that the two files be installed in the correct order, see below.

To install the WapIDE on Windows:

If WapIDE_3PP_2_0 is not installed then install it by running the `.exe` file.
This can be done in two ways:
– Choose **Run** from the Windows Start Menu and locate the `.exe` file by either enter the path to the file or by pressing **Browse.**
– Double click the .exe file directly.
Then follow the instructions in the installation program.

When WapIDE_3PP_2_0 is installed then install WapIDE_SDK_2_1 in the same way as for the 3PP part above.

The default directory for installations is:
`C:/Program Files/Ericsson/sdk`

Installation overview

By default an **Ericsson** menu is added to the Programs section of the Windows Start menu. The structure is:

Ericsson → WapIDE →	Readme	release notes
	WapIDE 20	the exe file
	WapIDE Help	help can be accessed from inside WapIDE or directly here
	WML server	starts the Xitami web server

Together with WapIDE some other parts are also installed: a Xitami Web server, sample examples and a support library. To find out more about the Xitami web server see "The Xitami web server". By default the Xitami web server is installed to start automatically at start up of your computer.

Uninstalling the WapIDE SDK

To uninstall the WapIDE use the Windows **Add/Remove** Program in the Control Panel folder in the **Settings** section of the Windows Start menu. Start with uninstalling the WapIDE and then the 3PP.

Note! Deleting the WapIDE files and directories manually won't completely uninstall it. You have to use the procedure above.

WapIDE files

Together with the WapIDE Program a number of directories and files is also installed. This section gives a brief overview of a typical WapIDE installation outlining the useful and/or important ones.

File or directory	Description
`Samples/`	Example applications created with WML and WMLScript. You can use these to get ideas and help for your own applications.
`SDK/`	The main directory for the SDK
`SDK/Docs/`	Release notes, Help files and Deviation descriptions
`Support/`	Information related to the other parts of the SDK environment: Erlang, Perl, tcl, web server, winsock2 and zoomin

Known problems

This is a short list of some problems that might happen when installing WapIDE and starting it. This is a list that can help you get WapIDE going if you have problem installing or starting WapIDE. For problems not covered in this document you can reach support at http://www.ericsson.com/developerszone.

error	**Solution**
Application designer can't enter test mode	See "Browser won't start"
Browser won't start	Are you running without a network connection? If you want to do that you have to set the IP address to a dummy value. This can be done from Settings/Network/Protocols/TCP IP. The dummy value can be anything that looks like an IP address.
Browser does not start the default home page.	The browser starts with "device_home.wml". Check to see that it exists and contains correct code. The path is: `../Ericsson/sdk/sdk/browser/erlang/device_home.wml`
Can not run Perl scripts form the Xitami server	You need to (re-)start the Xitami server from the Ericsson/WapIDE menu so that the path to the Perl library is set correctly.
"error: corrupt cabinet file"	The download files have not been downloaded correctly. Download them again.
"error: Could not open HTTP port. Port already used by other server"	You probably have another web server running on your machine A solution to this is to configure your computer so that only one of the web servers start automatically and to not have them running at the same time later either.
"error: out of environment space..."	The default allocation of memory for environment variables is probably exceeded. This usually helps: 1. Start Settings/Taskbar form the Start menu 2. Select Start Menu Programs and Advanced 3. Open Programs/Ericsson/WapIDE 4. Select WapIDE 2.1 and edit its properties 5. Under Memory change the Initial environment setting from AUTO to 4096 6. Start WapIDE again If this helps you should probably do the same with the WML Server menu entry in the WapIDE menu.
"error while autoloading "configDevice". Expected floating-point number but got "60.0""	There seems to be an error when using the browser v 2.0 with Swedish settings in NT environment. Set the NT environment to be "English UK"

Using the WapIDE

When you start WapIDE (by the menu Start/Programs/Ericsson/WapIDE) you will see a start panel that will look like the Figure 1. This start panel is used to access the included tools: the Browser, the Application Designer and the Server Toolset.

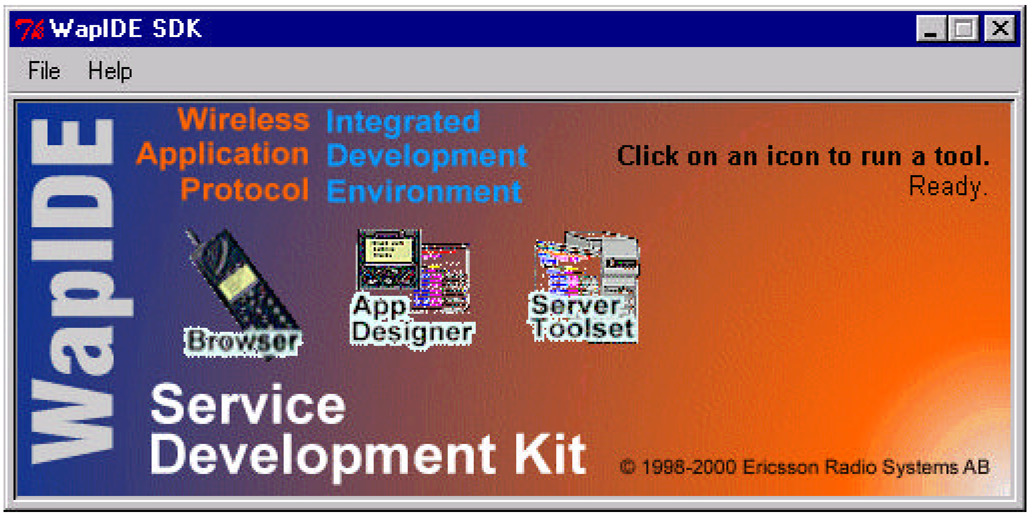

WapIDE SDK
File
Help
WapIDE
Wireless Application Protocol
Integrated Development Environment
Click on an icon to run a tool.
Ready.
Browser
App Designer
Server Toolset
Service Development Kit
© 1998-2000 Ericsson Radio Systems AB

Figure 1 WapIDE SDK main window

The WapIDE Browser simulates a WAP device and interprets cards, decks, and scripts written in WML. The Application Designer lets you create and test your own WAP applications. The Server Toolset contains various tools and library functions that support development of WAP applications. This includes WML and WMLScript compilers and Perl for creating dynamic WML source.

Browser

The Browser is used primarily to view applications, i.e. to browse WML cards. It also interprets WMLScript. The tool can also be used as a WML browser instead of using a WAP device to access WAP applications developed by you or others. It reads both binary WML bytecode and textual WML files.

There are three ways to load contents to the browser; from a:

- WAP gateway using WAP client stack. To be able to load the WML decks using WAP communication stack from a web server you must have access to a WAP gateway
- Web server using HTTP client
- File containing contents saved on the hard drive.

The Browser supports the design and use of applications in other languages with different character sets, such as for example Chinese.

Start the Browser

Start the tool by clicking the **Browser icon** in WapIDE main window. It is also possible to open a Browser via the **Load Device** menu in the File menu in the Browser window menu.

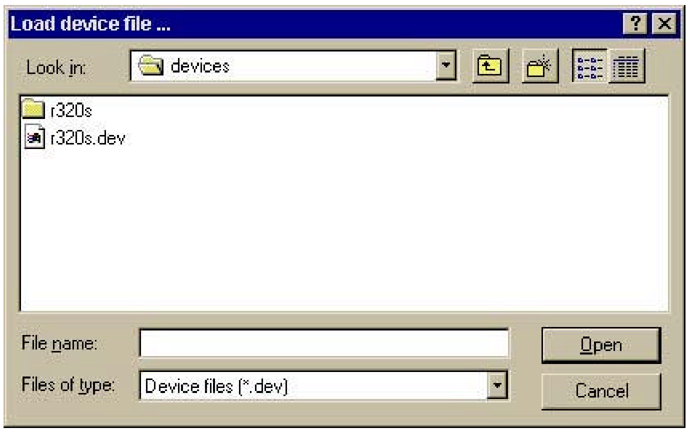


Figure 2 Browser: load a device

The window will resize for the loaded Browser. When the browser is started you can see the application in the display and use the buttons on the phone to enter commands and data.

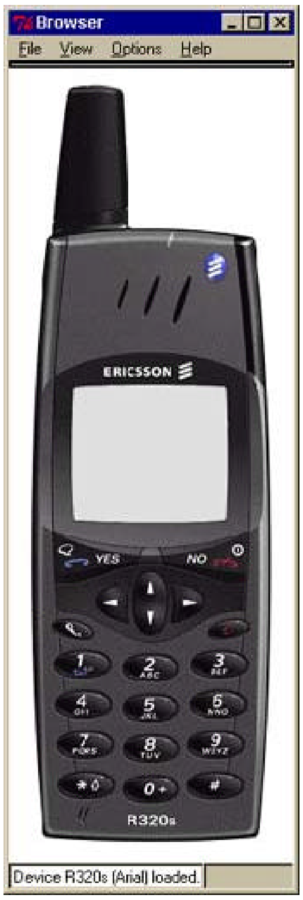


Figure 3 Browser: a R320 device is loaded

Changing the Appearance

To view the device without the outer window displayed, select **No decoration** from the View menu. In this format, use the right mouse button to use the menu options.

Locking/unlocking the Device

To lock/unlock the device position within the window select/de-select **Lock** from the View menu. When the device is unlocked, you may drag the device up, down, left, or right within the displayed window. To restore the device to the default position, select **Restore** from the View menu.

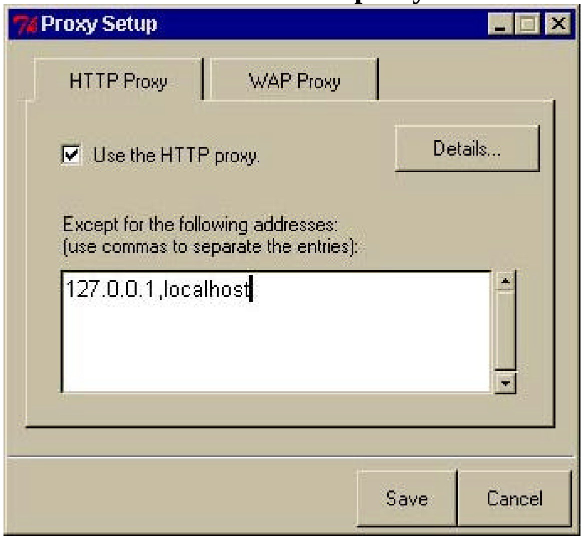
Application Access Setup

When accessing an application over HTTP or WAP protocol stack, some parameters must be configured. Also if a firewall is used, which generally is the case in a company's computer network, you have to make an HTTP proxy setup. If you are not placed behind a firewall there should not be need for HTTP proxy setup in WapIDE.

To set the parameters, go to the **Proxy Options** in the Options menu in the Browser window. A window will open for configuration for both HTTP and WAP Proxy. Follow the steps below.

HTTP Proxy

1. Go to the **HTTP Proxy** tab
2. Tick the box: **Use the HTTP proxy**.



[✓] Use the HTTP proxy.

Except for the following addresses:
(use commas to separate the entries):

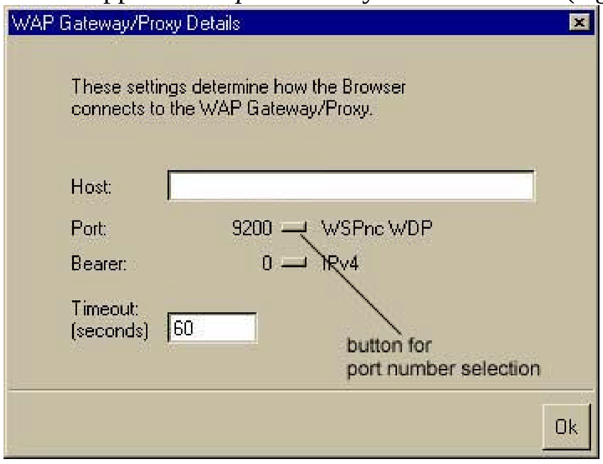
127.0.0.1,localhost

Figure 4 Browser: the setup window for both HTTP proxy and WAP Proxy

3. Press the **Details** button. Here you have to specify the hostname that should be used. This hostname is the same as for your Internet browser, e.g. `www.proxy.company.com`. Also set the port (normally port 80 for HTTP) and timeout (e.g. 60 seconds).
4. In the first Proxy setup window there is a list of **exceptions** for using the proxy. The list should contain at least two items, `localhost` and `127.0.0.1`, otherwise the sample applications might not work.

WAP Gateway/Proxy

1. Go to the **WAP Proxy** tab.
2. Tick the box: **Use the WAP Proxy**.
3. Press the **Details** button and write the IP address of the WAP Gateway. *Ericsson offers free access through a WAP GW/Proxy for registered Ericsson Developers' Zone users.* The IP address can be found in the Developers' Zone Test Area. Also select the WAP port to use by click-and-hold-down on the small button. The WAP port specifies to the WAP GW what WAP communication stack to use. There is also a button to select which bearer to use. Today the only bearer supported is *Ipv4*. Finally set the timeout (e.g. 60 seconds).


WAP Gateway/Proxy Details

These settings determine how the Browser connects to the WAP Gateway/Proxy.

Host:
Port: 9200 WSPnc WDP
Bearer: 0 IPv4
Timeout: (seconds) 60

Figure 5 Browser: the details window for WAP Proxy setup

4. If you want to test applications via the gateway on your local host you have to remove 'localhost' from the list of exception addresses.

If a WAP Proxy is used the Browser tries to connect first to the specified WAP proxy, and then to the HTTP proxy.

The **exception list** defines hosts where the routing will not go through the WAP or HTTP gateway, but route directly to the specified host.

Use the browser

As mentioned above there are three ways to load contents to the browser, from a WAP gateway using WAP client stack, a Web server using HTTP client or a file containing contents saved on the PC you are running WapIDE on. In all these cases you have to enter an URL to access the application you want. Before you can load an URL or do anything else you have to turn the device on. When the device is activated WapIDE automatically loads a welcome deck with links to some sample applications provided in the installation.

To turn on a device, select the **power button**. If you aren't sure which button it is, then select **Device Info** from the Help menu. There should be a hint on how to turn the device on.

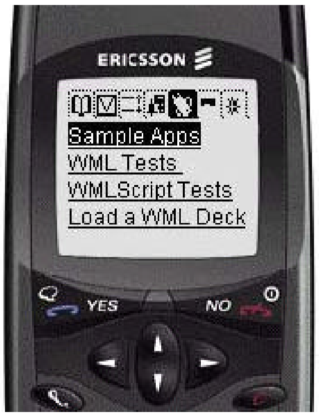
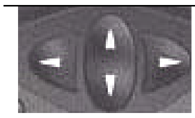


Figure 6 Browser: a started device

For *Ericsson phones* do the following: Turn the device on by pressing and holding the **NO** key. After the device is activated, the following generic actions applies for Ericsson devices:

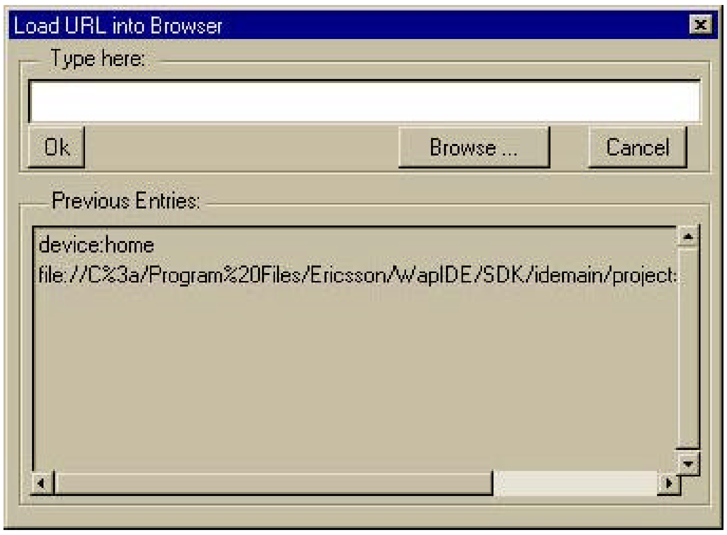
Button	User action	Device action
YES	Short click	Sends an ACCEPT event
	Hold-down click	Shows a list of options, including soft options
NO	Short click	Goes BACK in history
	Hold-down click	Turn device OFF
	Up / Left	Go up in links or selection list
	Down / right	Go down in links or selection list

For further information on the MMI of Ericsson WAP devices see **Device Info** in the Browser Help menu or download *Design guidelines* for different Ericsson WAP terminals, found at the *Ericsson Developers' Zone* at http://www.ericsson.com/developerszone.

Access an application

Applications are accessed by an URL. To load a URL, you must first be sure that the device is on, then select **Load URL** from the File menu. Now you can do one of the following:

- type in the address as a normal Internet URL, e.g.: `http://wap.fictional.ericsson.com/welcome.wml`
- select an URL from the field **Previous Entries**
- use **browse** to find a WML or a binary WML file at your computer



device:home
file://C%3a/Program%20Files/Ericsson/WapIDE/SDK/idemain/project:

Figure 7 Browser: load an application

The browser will load the application and show the first card in the display.

In Developers Zone there is a list of links to WAP enabled sites, you can try one of those.

Data entry

You can enter data in two ways:

- Directly into the browser by selecting an input field and then use the device buttons to type data in.

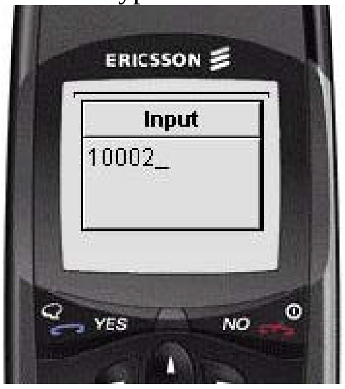
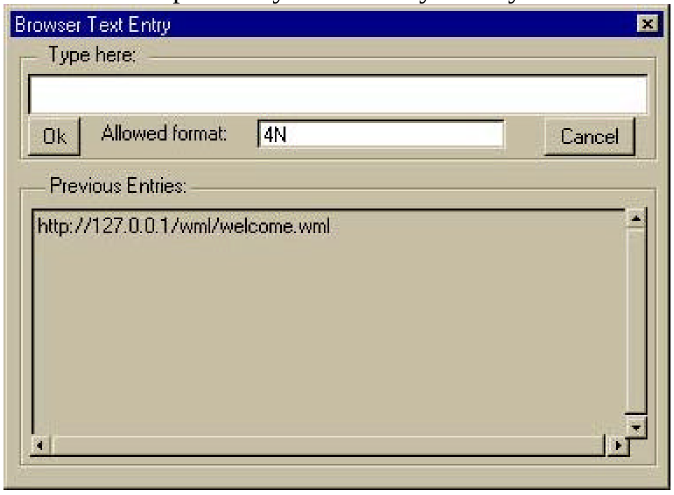


Figure 8 Browser: enter data in device window

- By right-click when the mouse courser is placed in the input field. An input window will open and you can use your keyboard to enter the data.



Allowed format: 4N

http://127.0.0.1/wml/welcome.wml

Figure 9 Browser: enter data in text window

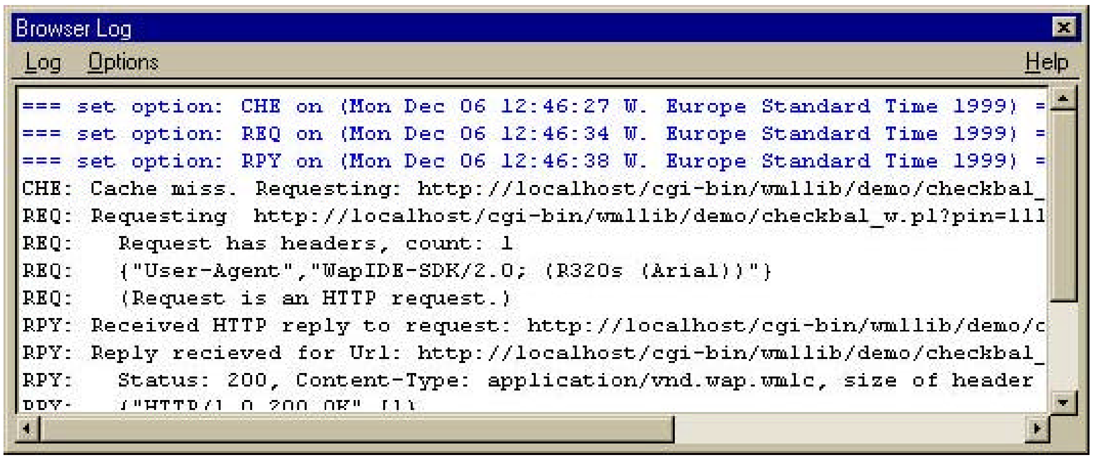
This data entry window can be used also when entering data in another type of character set.

Note! You cannot turn the device off if you are in input mode.

Browser Log

The Browser Log is a tool for tracing. Here you can see what data is sent, if the WAP stack or HTTP is used, if the application is fetched from cache and what response you get back.

The Browser Log is opened from the Browser window, go to the View menu and choose **Log Window**. All logs are ‘off’ when the log window is opened. In the **Options** menu in the Log window, you can choose whether all logs, one or you selection of logs are to be shown. Under the **Log** menu in the Log window, you can clear, save and close the log window. Help about the Browser Log can be reached from the **Help** menu in the Log window.



```
=== set option: CHE on (Mon Dec 06 12:46:27 W. Europe Standard Time 1999) =
=== set option: REQ on (Mon Dec 06 12:46:34 W. Europe Standard Time 1999) =
=== set option: RPY on (Mon Dec 06 12:46:38 W. Europe Standard Time 1999) =
CHE: Cache miss. Requesting: http://localhost/cgi-bin/wmllib/demo/checkbal_
REQ: Requesting  http://localhost/cgi-bin/wmllib/demo/checkbal_w.pl?pin=111
REQ:   Request has headers, count: 1
REQ:   {"User-Agent","WapIDE-SDK/2.0; (R320s (Arial))"}
REQ:   (Request is an HTTP request.)
RPY: Received HTTP reply to request: http://localhost/cgi-bin/wmllib/demo/c
RPY: Reply recieved for Url: http://localhost/cgi-bin/wmllib/demo/checkbal_
RPY:   Status: 200, Content-Type: application/vnd.wap.wmlc, size of header
RPY:   {"HTTP/1.0 200 OK" [1]}
```

Figure 10 Browser: the log window

Note! To turn off the log functionality you have to close the Log window by the menu **Close** in the menu Log in the Log window. Closing the Log window by the "X" button in the upper right corner will not turn off the logging.

If you have the log active and access an application containing pictures the log might interpret the picture data as control characters and malfunction. To avoid this problem make sure that the log functionality is turned off when accessing pictures.

Log actions	
Clearing a Log	To clear the log window of all data, select Log, and then **clear** from the menu items.
Saving a Log	To save a log, select Log, and then select **save** from the menu items. You will need to select a directory/file to save your log to.
Cache Logging	To cache the log, select Options, and then select **cache** from the menu items. The trace indicates cache hits, misses, and clearance.
Request Logging	To trace WML replies, select Options, and then select **Request** from the menu items.
Reply Logging	To trace WML requests, select Options, and then select **Reply** from the menu items.
Source Logging	To log your WML source, select Options, and then select **Source** from the menu items. The source will be logged when a new URL is retrieved, or a card in the deck is switched.
WML/WAP Logging	To trace WML and WAP events, select Options, and then select **WML/WAP** from the menu items.
Timer Logging	To view the Browser related timer events select Options, and then select **Timer** from the menu items.
Device Logging	To view the Browser related device events select Options, and then select **Device** from the menu items.

Variable trace

It is possible to trace on WML variables and values during execution in the Browser. This is done in the WML variable viewer window that is initiated from the Browser, go to the **View** menu and choose **WML Variables**, a window will start showing the current variables and their values.

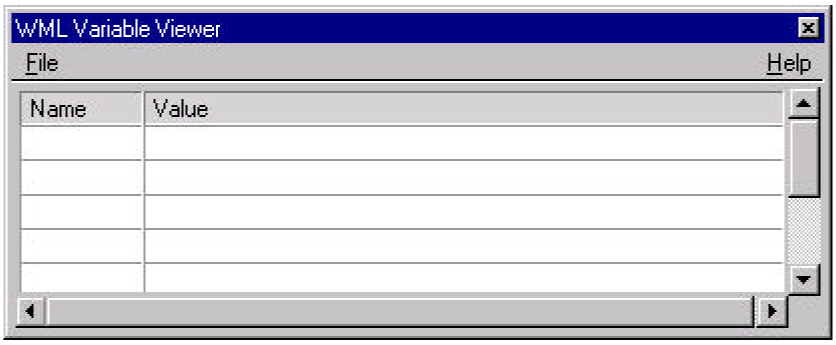


Figure 11 Browser: WML Variable Viewer

Browser preferences

Note! These values are set by default and typically should not be changed.

The engine port & host are for internal communication with the communication stack. The help browser search path points to the HTML browser used to browse help texts. The telephony modem device is used for WTA telephone access.

Note! The telephone modem device is only used for simulated WTA capabilities.

Help

For further help use the on-line help, which is reached by choosing **Help** in the Browser window and then **Browser**.

WML in other character sets

It is possible to use character sets other than ASCII in the browser. You can enter the encoding representing the characters directly, by typing the HEX representation in the browser or use an additional tool as for instance NJStar and then do a copy/paste action to enter the data into the browser.

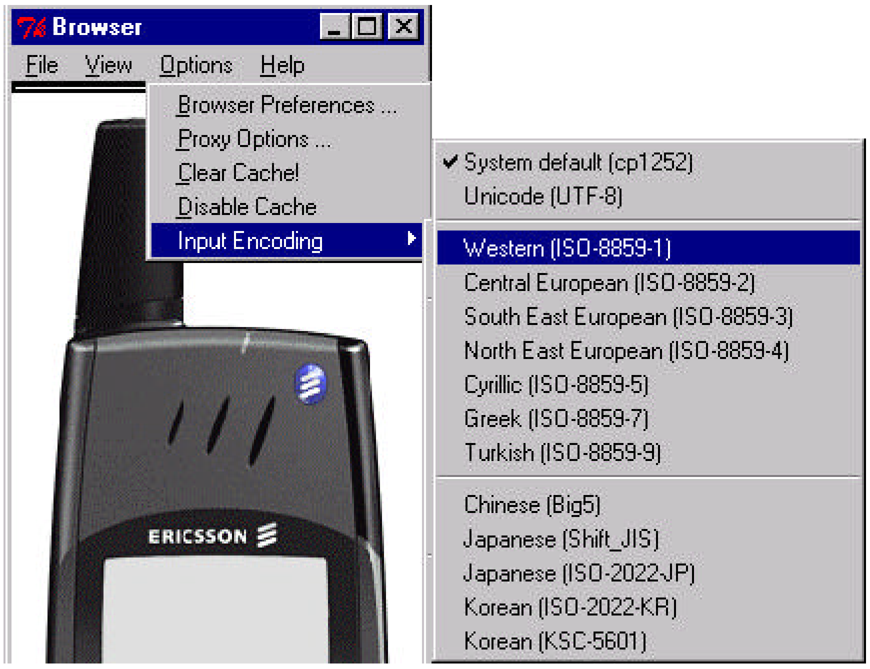


Figure 12 Browser: Supported character encoding

The browser converts the input shown on the screen into UTF8. The data managed by the Erlang process remains in UTF8 format. In addition, The **input encoding** for the browser should be set from the options menu to ensure a right interpretation of the entered data.

It is important that you have an appropriate font installed on your system if you want a particular character to be displayed. The browser uses the mechanisms on TCL/TK 8.1 to locate the correct font on you system. In some cases the browser is unable to identify a suitable font, in which case the character can not be displayed. Instead the browser will display a system dependent fallback character such as "?" or " ".

Application Designer

Using the tool Application Designer you can write and test WML and WML Script code and see the result of the coding. A WML deck can be either of two types, static or dynamic. The content of a static page is always the same, while the content of a dynamic page changes from time to time. The Application Designer only supports creation of static pages. To make dynamic pages you can use the libraries wmllib (see section about Server Tools) or JAFFA (Java Servlet Library, this tool can be downloaded from Developers Zone).

The application designer includes a WML editor with syntax highlighting and integrated compilers for WML and WMLScript and a fully featured browser for running the compiled application just by switching the application designer into *test mode*. This browser is the same as the stand-alone browser tool.

Start the Application Designer

Start the tool by clicking the **Application Tool** icon in the WapIDE main window. You will then get a window where you can choose to create a new WML application project (see ”Creating a New Project” for details) or load an existing WML application project. A project consists of a WapIDE device and a number of files that contain f.e. WML code. The device will be used to interface the browser process so that the user may test the WML decks directly in the application designer.

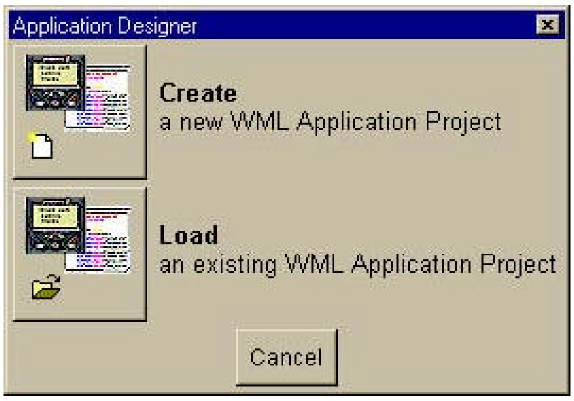
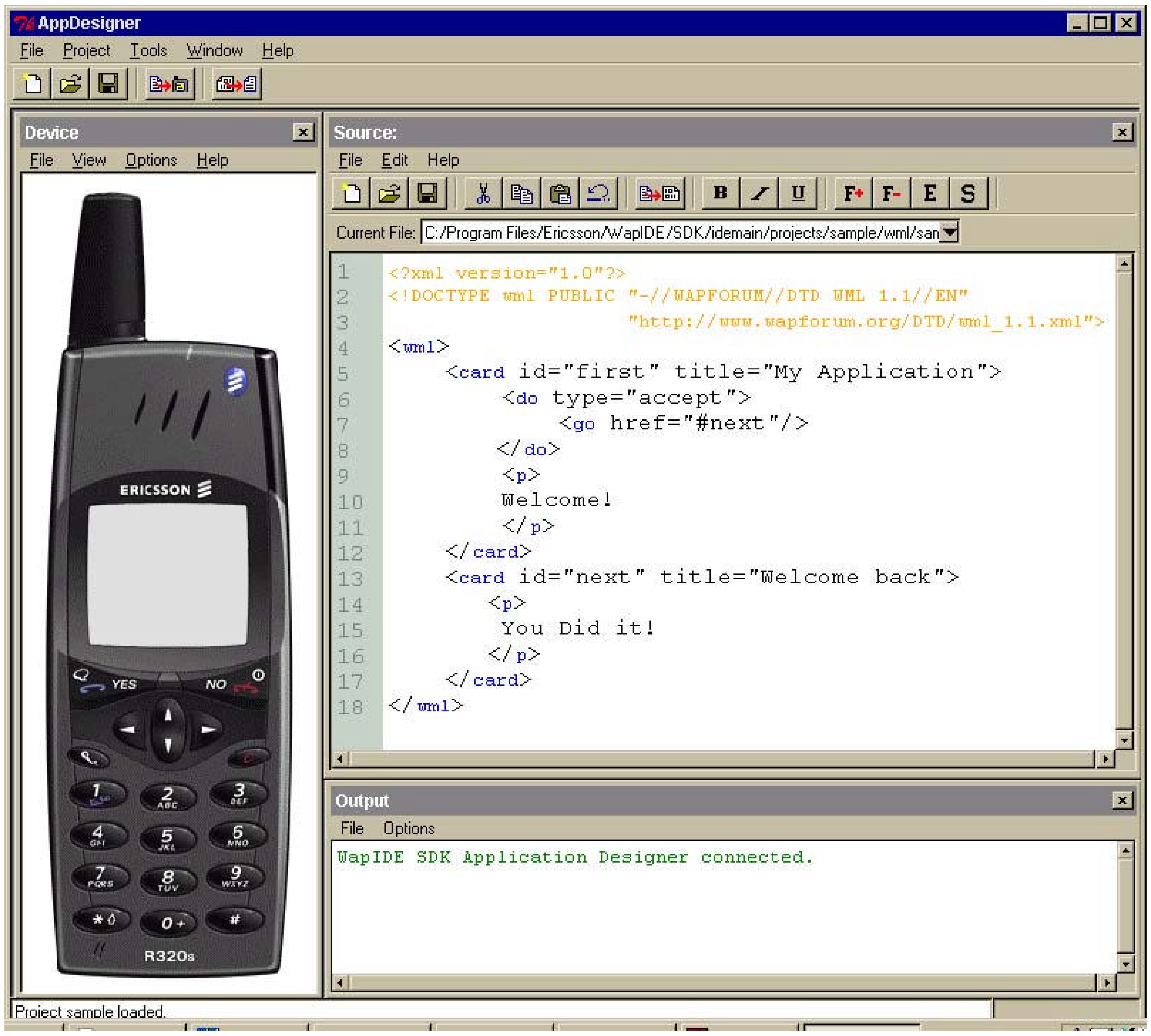

Application Designer
Create
a new WML Application Project
Load
an existing WML Application Project
Cancel

Figure 13 Application designer: start up window

The application designer has three windows:

- Device
- Source
- Output.

In the **Source** window you enter the WML or WML Script code, in the **Output** window you see the compilation results and in the **Device** window the browser, with which you can test you application. If one of the windows is not opened by default, it can be opened or closed by the **Window** menu in the main application designer window.



Current File: C:/Program Files/Ericsson/WapIDE/SDK/idemain/projects/sample/wml/san

```
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.1//EN"
                     "http://www.wapforum.org/DTD/wml_1.1.xml">
<wml>
    <card id="first" title="My Application">
        <do type="accept">
            <go href="#next"/>
        </do>
        <p>
        Welcome!
        </p>
    </card>
    <card id="next" title="Welcome back">
        <p>
        You Did it!
        </p>
    </card>
</wml>
```

```
WapIDE SDK Application Designer connected.
```

Project sample loaded.

Figure 14 Application designer: main window with all three use windows active

Create or edit a WML application project

When starting the Application designer you will be asked whether to open an existing project or start a new one (see “Start the Application Designer”). You can also start or open a project by using the file menu in the Application designer main window, and then select open project/new project from the menu items. WapIDE uses projects to keep together WAP applications. A project can consist of one or several WML or WMLScript files.

Creating a New Project

1. fill in the project name,
2. accept or change the location where it will be stored
3. choose the device to test the applications in the project on. (This can be changed later on.)

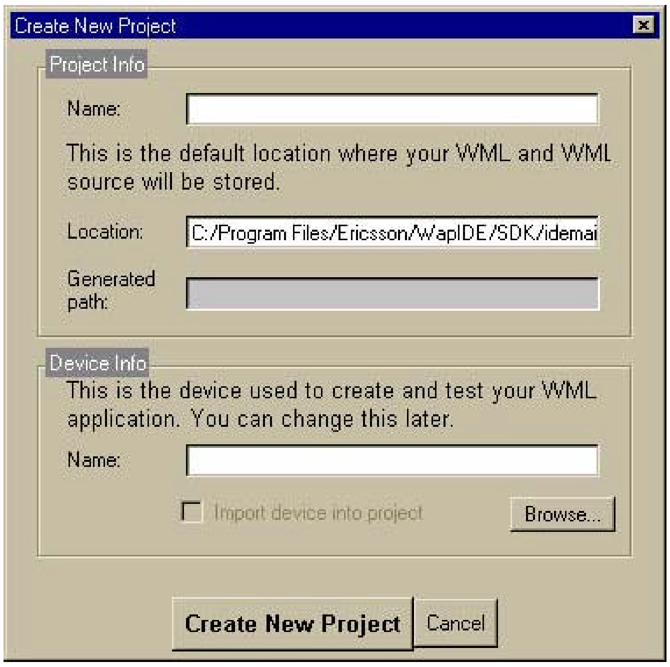

Create New Project
Project Info
Name:
This is the default location where your WML and WML source will be stored.
Location:
C:/Program Files/Ericsson/WapIDE/SDK/idemai
Generated path:
Device Info
This is the device used to create and test your WML application. You can change this later.
Name:
Import device into project
Browse...
Create New Project
Cancel

Figure 15 Application designer: create a new WAP application project

The default location for created projects is:
`C:/Program Files/Ericsson/sdk/SDK/idemain/projects`

The project folder that is created consists of some sub-folders. These are:

- `cgi-bin`, to store scripts to create dynamic applications
- `device`, to store device information if not the standard device supplied with WapIDE
- `wml`, to store WML and WMLScript code

Also created is a project file placed in the project folder and with the extension `.wap`. It stores information such as which device is connected to the project, the last WML and WMLScript file to be opened etc. It should **not** be edited by hand.

Create WML / WMLScript

The Source window is used for creating WML or WMLScript code. After you have opened or created a Project a new code file is opened in the source window. The code can be *compiled* for error checks and can also be *tested* in the Device window continuously. WML files have the extension `*.wml`; WMLScript files have the extension `*.wmls`.

Create or edit WML Code

To create a new WML file *outside an existing project* select **New Project** in the File menu in the Application Designer main window and fill in the project and device information just as when creating a new project.

To create a new WML file *inside an existing project* select **New** in the File menu in the Source window and an empty file will open in that window.

To create or edit WML code, place the cursor inside the source window where you want to enter new code. You can then either fill in the code manually or use the *automatic WML tag insertion* function. This is accessed by the **Insert** menu in the

Edit menu in the source window. You can also access the insert menu by right-clicking your mouse. The automatic insertion menu is context sensitive and only allows insertion of tags proper to the flow in the code. If the WML element you have selected has attributes a dialog box will appear for you to set the values. Only the attributes marked with a star are compulsory. The others may be left out.

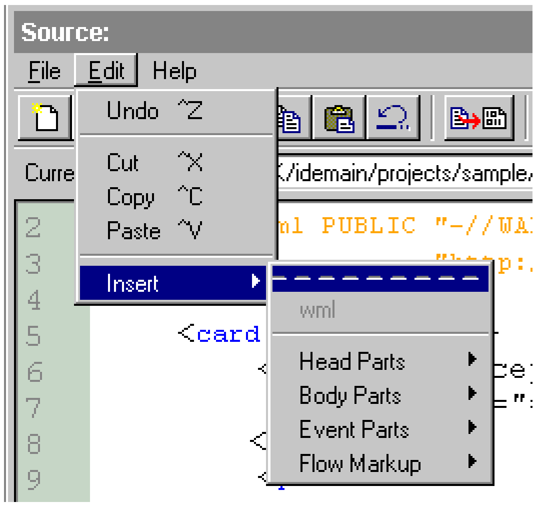


Figure 16 Application designer: the tear off menu for insertion of WML tags

The Insert menu is a 'tear off' menu which means that you can get the menu as a separate toolbar. You do that by going to the Edit menu in the Source menu, place the cursor on the dashed line, see Figure 16, and then release the mouse button.

The source window also has some predefined action buttons to help with some common tasks in constructing WML:

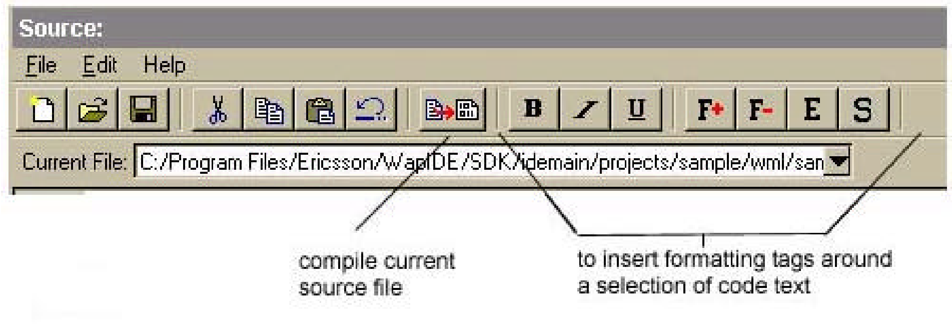

Source:
File
Edit
Help
Current File: C:/Program Files/Ericsson/WapIDE/SDK/idemain/projects/sample/wml/san
compile current source file
to insert formatting tags around a selection of code text

Figure 17 Application designer: action buttons in the Source window

WML tags inserted by the Insert menu or other built-in code creation help in the Source window will be color coded. WML tags inserted manually by you will not be color coded until you have performed a **save** and then a new **open** on the source file.

Create or editing WMLScript code

To create a new WMLScript file *inside or outside an existing project* do in the same way as for "Create or edit WML Code".

The tag insertion help in the Source window is only for WML code. To create or edit *WMLScript* code the Source window should be considered as an ordinary text editor. You can use the Source window to open an existing WMLScript file for editing or create a new script file.

As help in creating WMLScript WapIDE comes with a number of sample WMLScript files. These can be found in the installed folder:
`C:/Program Files/Ericsson/sdk/Samples/wml/testscripts`

Create applications in other character sets

It is possible to use character sets other than ASCII in the application designer. You can enter the encoding representing the characters directly, by typing the HEX representation in the Source window or use an additional tool as for instance NJStar.

By default the application designer adds an encoding attribute that specifies latin1 (ISO-8859-1) to the XML declaration. If you want to write your application using another document encoding, you need to remember to change this attribute (or remove it and set the "*Content-Type*" transport header).

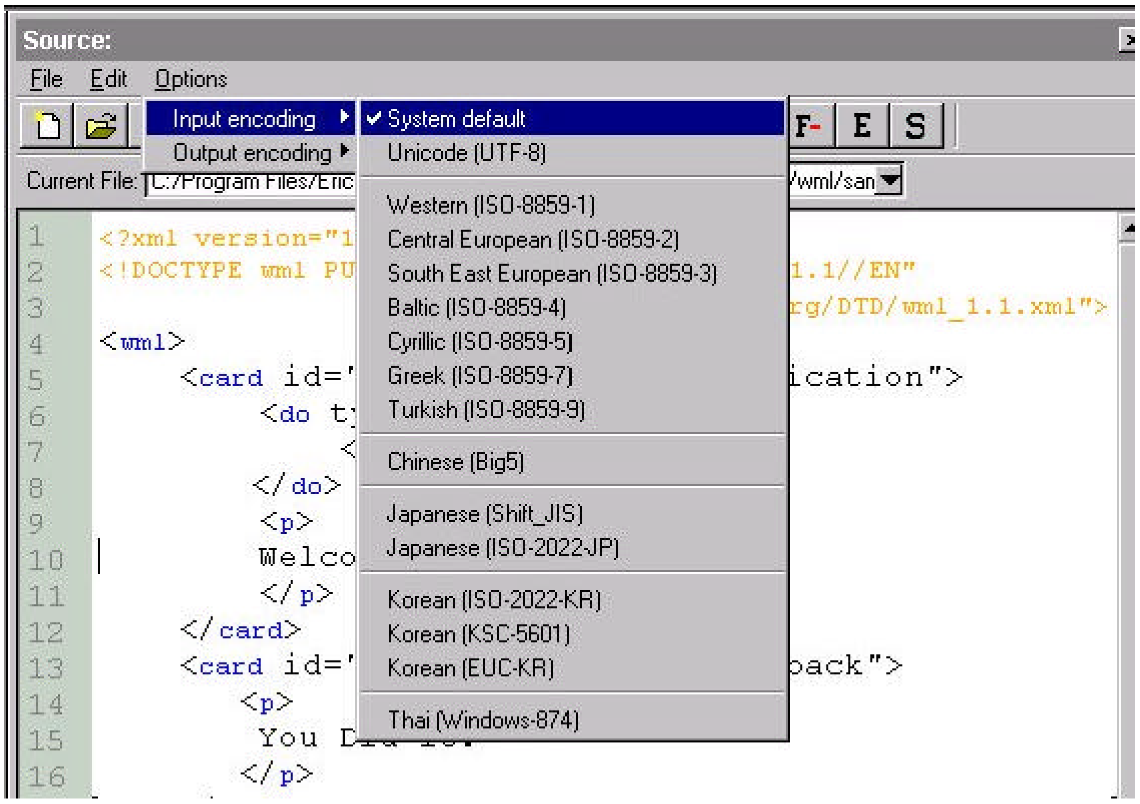


Figure 18 Application designer: List of output encoding

You may also convert the current document encoding, e.g. from Thai (Windows-874) to Unicode (UTF-8). To do this you need to change the encoding settings in the **Options** menu. The **Input Encoding** should be set to Thai and the **Output Encoding** to Unicode. Then, when you save the document the content will be reloaded in the new encoding, and with the encoding attribute properly altered.

Compile and test applications

You may compile or test files at anytime when you are creating or editing your application.

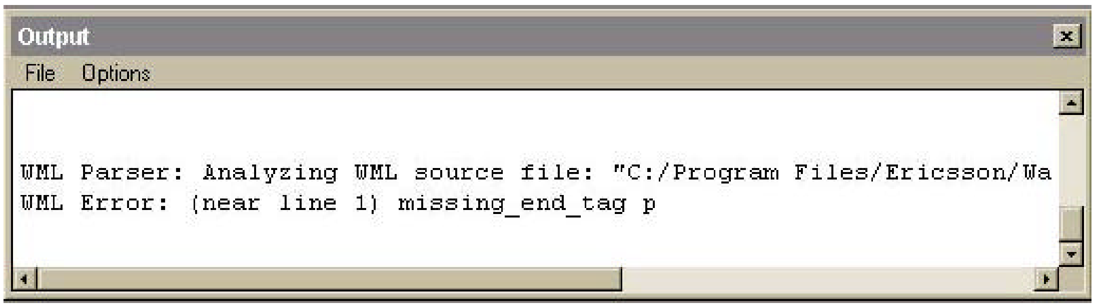
Compiling

To compile a WML or WMLScript file you can:

- select the **compile button** from the Source window
- select the **compile menu** from the file menu in the Source window

When compiling WapIDE will react to the file extension (`*.wml` for WML, `*.wmls` for WMLScript) and invoke the right compiler. Therefore it is important to save the file with the right file extension.

Syntax errors in the code that is detected will be presented in the Output window:

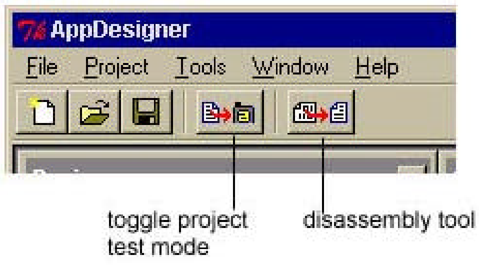


```
WML Parser: Analyzing WML source file: "C:/Program Files/Ericsson/Wa
WML Error: (near line 1) missing_end_tag p
```

Figure 19 Application designer: compile log in the Output window

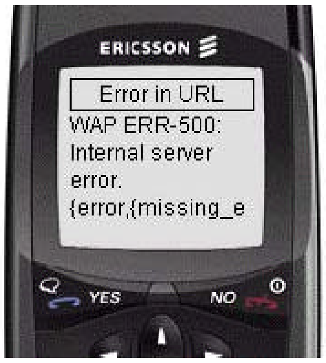
Testing

You can continuously test your application in the Device window. It is possible to test without compiling the code first. To test your WAP applications select the **Test** menu in the project menu in the Application designer window. Alternatively, you may use the **toggle** button in the Application designer window for the test mode.



When the Application designer is in test mode you can not edit the application code in the Source window. To be able to edit the code you have to toggle back to from the test mode.

Syntax error in your code can be detected in the testing mode by the Device display showing an error code:



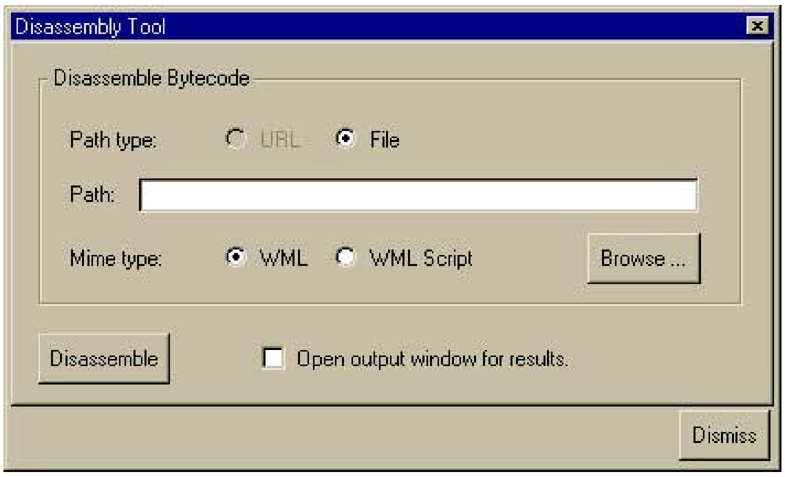
```
Error in URL
WAP ERR-500:
Internal server
error.
{error,{missing_e
```

Figure 20 Application designer: a syntax error detected in the device

However we recommend that the compile function be used as the primary tool to find syntax errors, since the log is more informative.

Disassembly

A WML or WMLScript bytecode disassembler is also available as a debugging tool. It can be used to load a *compiled bytecode* file and get a translation of the bytecode. Select the Tools menu in the Application designer window and then **Disassemble** to start the disassembler. The results can be viewed in the Application Designer Output window.



Path type: ( ) URL [✓ File]

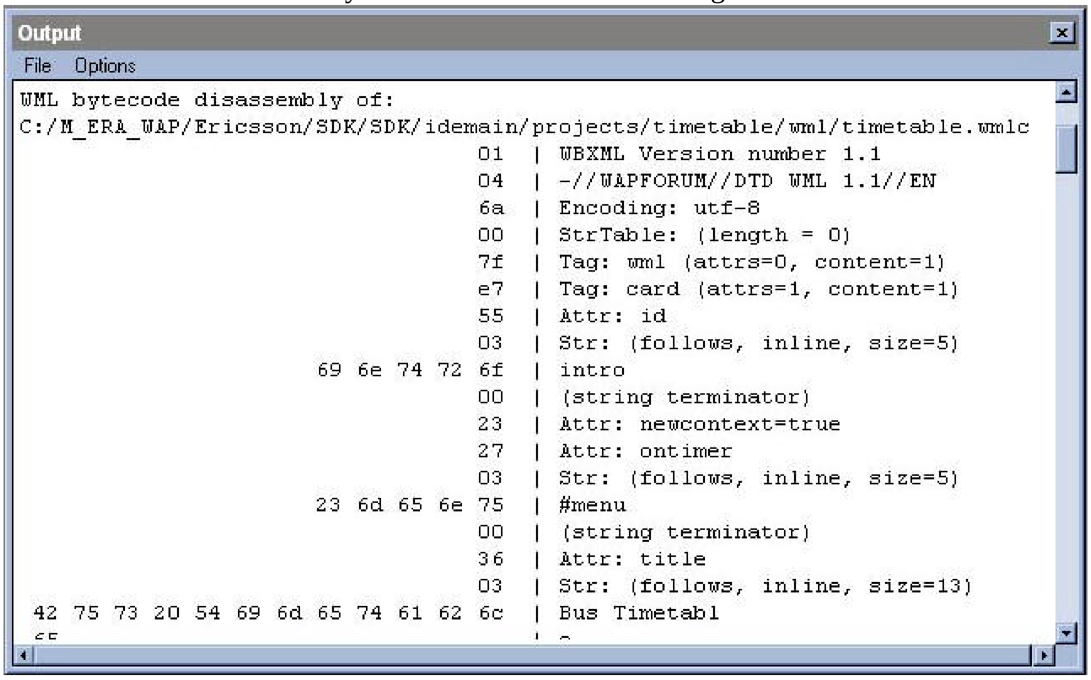
Path:

Mime type: [✓ WML] ( ) WML Script

☐ Open output window for results.

Figure 21 Application designer: disassembly of compiled code

A disassembled bytecode file can look like the figure below.



```
WML bytecode disassembly of:
C:/M_ERA_WAP/Ericsson/SDK/SDK/idemain/projects/timetable/wml/timetable.wmlc
                                   01  | WBXML Version number 1.1
                                   04  | -//WAPFORUM//DTD WML 1.1//EN
                                   6a  | Encoding: utf-8
                                   00  | StrTable: (length = 0)
                                   7f  | Tag: wml (attrs=0, content=1)
                                   e7  | Tag: card (attrs=1, content=1)
                                   55  | Attr: id
                                   03  | Str: (follows, inline, size=5)
                       69 6e 74 72 6f  | intro
                                   00  | (string terminator)
                                   23  | Attr: newcontext=true
                                   27  | Attr: ontimer
                                   03  | Str: (follows, inline, size=5)
                       23 6d 65 6e 75  | #menu
                                   00  | (string terminator)
                                   36  | Attr: title
                                   03  | Str: (follows, inline, size=13)
 42 75 73 20 54 69 6d 65 74 61 62 6c  | Bus Timetabl
```

Figure 22 Disassembled bytecode

Help

For further use of the Application Designer use the on-line help found in the **Help** menu in the AppDesigner, Device and Source window respectively.

Server Tools

The Server Toolset can be used as a help to control your WML/WMLScript code and to create dynamic WAP applications. It contains various tools and library functions that support development of WAP applications. This includes WML and WMLScript compilers, Perl library with functions for creating WML source, and WML verifier. It also contains a set of sample applications.

The WML bytecode and WML script compilers are the same as in the Application Designer. The WML Syntax Analyzer is used to verify WML, which means the code is compared to the DTD (Document Type Definition), which is the XML-based definition for WML.

The Server Tools also contains a Perl library, `wmllib`. The functions in the wmllib make it easier to create dynamic pages. The library is found in
`C:/Program Files/Ericsson/Sdk/servtools/wmllib`
and contains documentation and examples.

The Server Tools also contains *sample applications* (which are the same as shown in the browser when starting it). The sample applications are found in
`C:/Program Files/Ericsson/sdk/Samples`
that contains two catalogues:

- `wml`, for the static WML and WMLScript.
- `cgi_bin`, for the dynamic parts that are implemented in using `wmllib`.

Start the Server Toolset

Start the tool by clicking the **Server Toolset** icon in the WapIDE main window. You will then see an information window where you can launch one of the toolsets:

- WML bytecode compiler
- WMLScript compiler
- WML 1.1 syntax analyzer

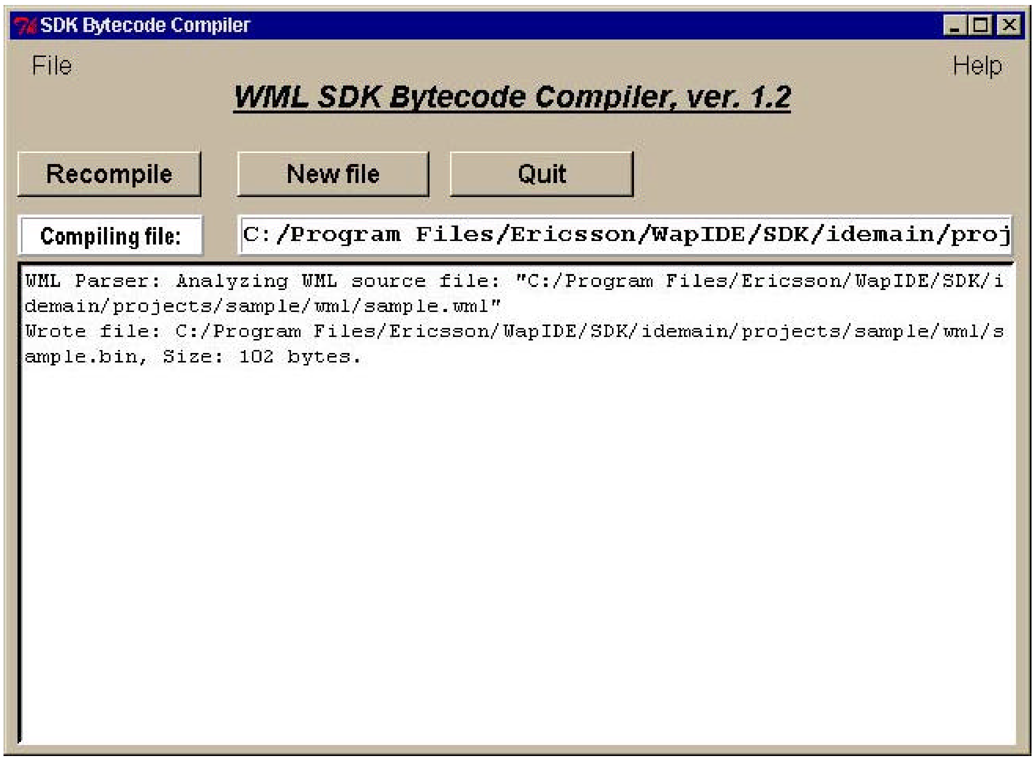
For further information see the respective sub chapters below.

Use the WML bytecode compiler

Compiled WML bytecode is the format used to and from the WAP client. In a typical use case the WAP GW receives text-based WML and compiles it before sending the data over a network to WAP clients that have built in interpreters for presenting the information. The main purpose of this tool is to:

- get the number of bytes of a compiled version of the WML file, and
- find any errors in the compilation. If there are any errors they will show in the compilation window in stead of the byte count.

To launch the WML bytecode compiler, select this item from the launch menu. This is a tool for compiling WML files. On starting the tool a window will open for you to browse to the file that is to be compiled. The result is shown in the compiler window.


SDK Bytecode Compiler
File
Help
WML SDK Bytecode Compiler, ver. 1.2
Recompile
New file
Quit
Compiling file:
C:/Program Files/Ericsson/WapIDE/SDK/idemain/proj

```
WML Parser: Analyzing WML source file: "C:/Program Files/Ericsson/WapIDE/SDK/i
demain/projects/sample/wml/sample.wml"
Wrote file: C:/Program Files/Ericsson/WapIDE/SDK/idemain/projects/sample/wml/s
ample.bin, Size: 102 bytes.
```

Figure 23 Server tools: the bytecode compiler

A new file to be compiled can be opened from the File menu or the button **New File**. After a file has been selected (which is shown in the `Compiling File` line) compilation is started by either pressing the button **Recompile** or from Compile in the File menu. The first time the source file is loaded the compilations is started automatically.

For a list of WML bytecode compilation errors, see **Help** in WapIDE main window.

Use the WMLScript compiler

Compiled WMLScript code is the format used in the WAP client. In a typical use case the WAP GW receives text-based WMLScript and compiles it before sending the executable file on to the client.

To launch WMLScript Compiler, select this item from the launch menu. On starting the tool a window will open to browse to the file that is to be compiled. The result is shown in the compiler window. This compiler generates code based on UTF8 or UCS2.

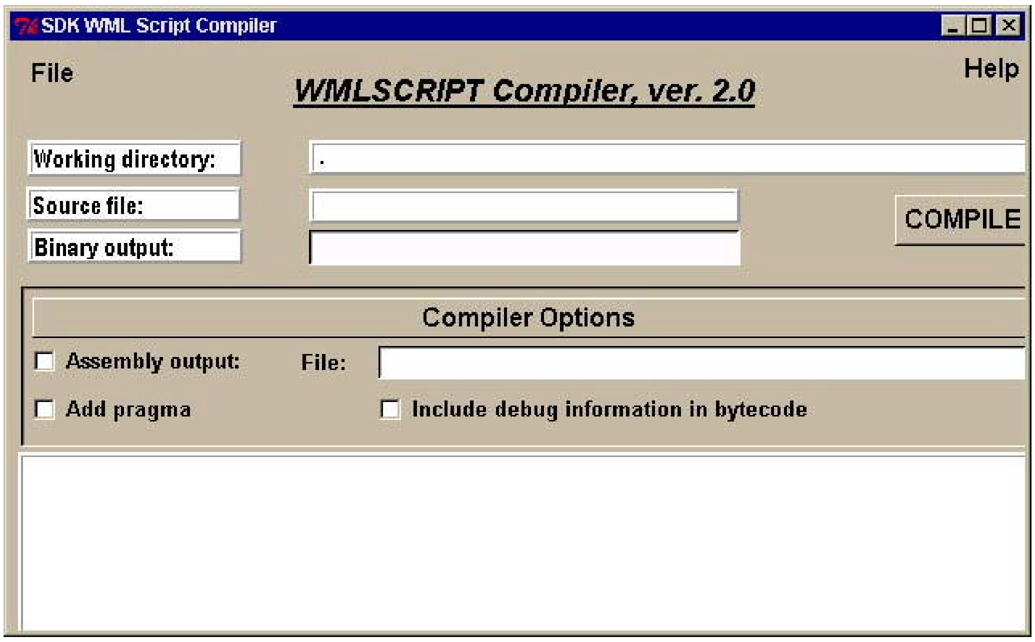


Figure 24 Server tools: the WMLScript compiler

There are some options within the compiler dialog:

1 **Assembly output** – this indicates if assembly code will be generated.
2 **Add pragma** – this indicates if pragmas are generated in the output or not. This can disable different access pragmas so testing can be run.
3 **Include debug information** – this indicates if debug output will be included in the compiled bytecode.

The use of *pragmas* is specified in the WMLScript specification from WAP Forum. The pragmas “specify compilation unit level information”. They have to be specified before the compilation starts. An example of a pragma is “use URL” which is used to give an alias for the URL to another WMLScript file where a function is stored that the complied script file calls.

You will need to press the **Compile** button or select Compile from the File menu to actually start compiling. This compiler generates code based on UTF8 or UCS2.

The output will indicate if compilation was completed without errors. If errors were found, there will be information about the error and an indication of the possible place in the code.

Use the WML syntax analyzer

The WML Syntax Analyzer is used to verify WML 1.1, which means the code is compared to the DTD (Document Type Definition), which is the XML-based definition for WML.

To launch WML syntax analyzer, select this item from the launch menu. On starting the tool a window will open to browse to the file that is to be analyzed. The output will indicate if the analysis was completed without errors, or if errors were found. The output will hold information about the error and on the possible place in the code.

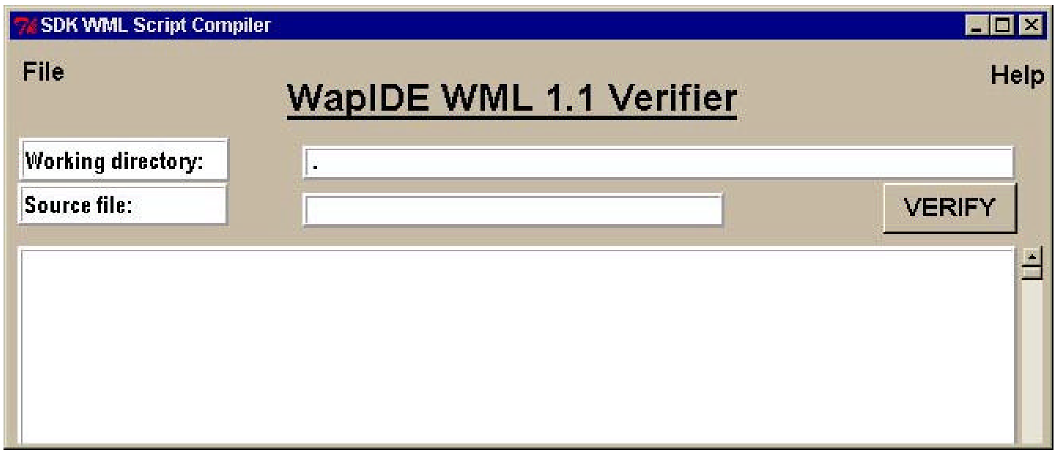


Figure 25 Server tools: the WML verifier to the WAP DTD

A new file to be analyzed can be opened from the File menu. After a file has been selected (which is shown in the `Source File` line) verification is started by either pressing the button **VERIFY** or from Verify in the File menu.

Note! The code can still contain errors since this is only verification against to the DTD and not a semantic control.

Perl libraries

Use the Perl library `wmllib.pl` to generate dynamic WML content. This library contains functions to generate WML content according to the WML Specification 1.1. It has been organized according to WML functionality. For more information on how to use the Perl libraries see **Help** in WapIDE.

CGI scripts

The Perl scripts found in the folder:

```
C:/Program Files/Ericsson/sdk/Samples/
      cgi-bin/wmllib/demo
```

are used in conjunction with the WML code to implement the demo applications that come with WapIDE. They make use of the `wmllib.pl` library of functions. This Perl library allows you to produce WML output from the CGI scripts.

The scripts are called from WML files containing a deck of one or more cards. The scripts will execute, do their calculations, fetch HTML pages, or whatever needs to be done. They return dynamically created WML decks and cards.

For further information about Perl libraries see **Help** in WapIDE. There you can also see the examples that are included in WapIDE.

Demo applications

A set of demo applications are implemented and delivered with WapIDE, and are as follows:

- Banking
- Stocks
- Tickets
- Black Jack

- Currency

For details on the sample applications and other examples of WML code se **Help** in WapIDE.

To set up a WAP application environment

The Xitami web server

The Xitami web server is an independent product from Imatix and is included to make it easy to get started with the sample applications included in WapIDE. All demo applications that come with WapIDE use the Xitami server.

If you have another web server or more than one instance of Xitami running at your computer you can get a problem with the port usage. The normal port number is 80 for web servers. Only one active web server can use a specific port number.

In the Windows Taskbar you can se a small Xitami icon to the right for every Xitami server you have running. The icon is a small green or red circle with a gray 'X' over. A green circle means that the web server is active.

Start and stop the Xitami web server

By default the Xitami web server is installed to start automatically at start up of your computer. You can start the web server manually at Start/Programs/Ericsson/WapIDE/WML Server

To stop the web server right-click on the icon in the taskbar and select `Terminate`.

To find out more

To find out more about Xitami you can go to:
`http://www.imatix.com/html/xitami/index.htm`

Configure a web server

If you want to place your WAP applications on a web server you have to configure it to support the right MIME types. To find out how to set the MIME types we refer to the manual of you server.

Content type	MIME type	File extension
WML source	text/vnd.wap.wml	wml
WMLScript	text/vnd.wap.wmlscript	wmls
Compiled WML	text/vnd.wap.wmlc	wmlc
Compiled WMLScript	text/vnd.wap.wmlscriptc	wmlsc
Wireless Bitmap	image/vnd.wap.wbmp	wbmp

Tutorial

To help you get started with creating applications with WapIDE we have included a guide how to create a new application all the way to testing it in the browser in WapIDE. In this chapter we assume that you have a basic knowledge of WML and WMLScript. We also assume that:

- WapIDE is installed
- The PC has a connection to the Internet
- The PC has Xitami web server installed (included in WapIDE installation)

For information on how to perform the actions not described in detail, we refer to the other chapters in this User Guide.

For design guidelines on how to create an application with good conformance to Ericsson WAP Products see reference list in "Related documents".

Creating your first WML application

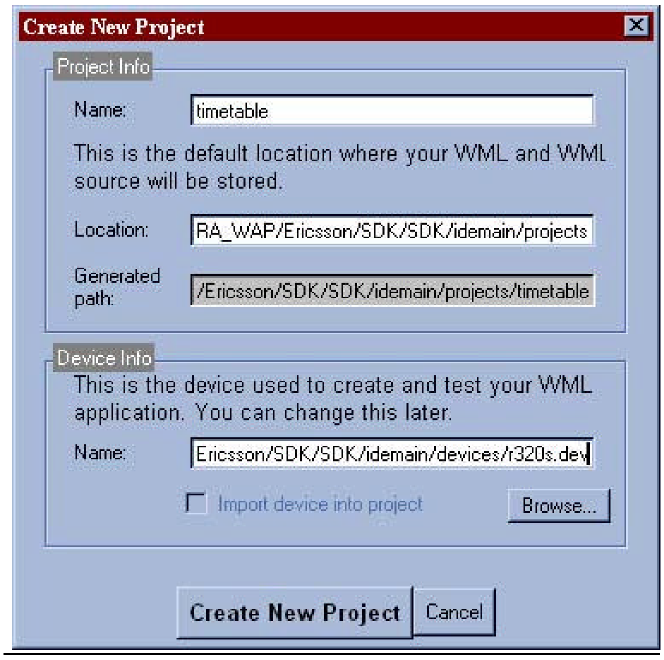
We will create a simple static WML file that is the first deck in a bus timetable application.

1 Start WapIDE

2 Start the Application Designer

3 Select "**Create** a new WML Application project"

Name the project `timetable` and set a device to test the WML code with: `r320s.dev` and press "**Create new project**". A notice window will inform that the new project was saved in the specified place. (If you want another place than the default place you must enter that before pressing the button.)



Name: timetable

This is the default location where your WML and WML source will be stored.

Location: RA_WAP/Ericsson/SDK/SDK/idemain/projects

Generated path: /Ericsson/SDK/SDK/idemain/projects/timetable

This is the device used to create and test your WML application. You can change this later.

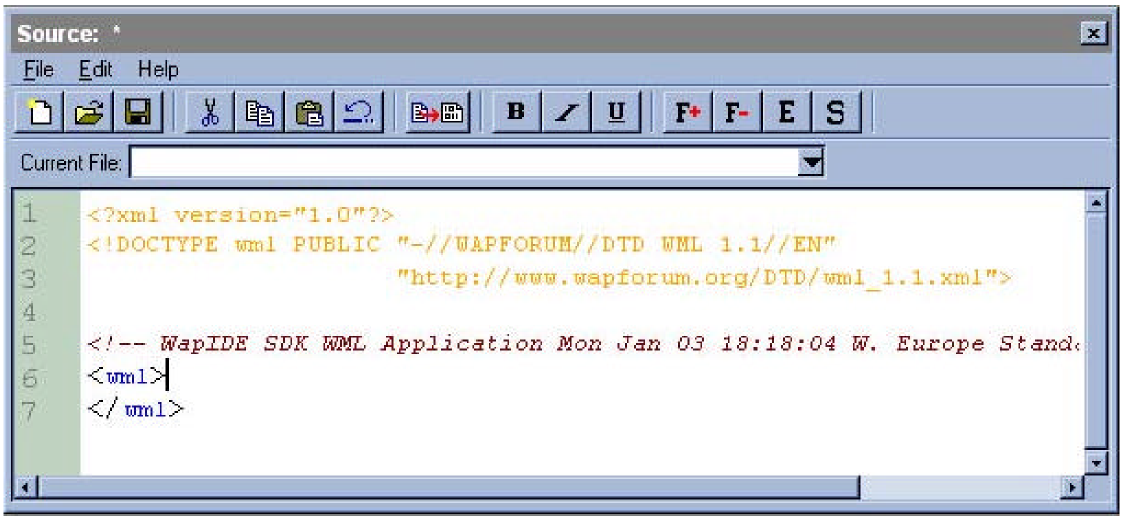
Name: Ericsson/SDK/SDK/idemain/devices/r320s.dev

☐ Import device into project

Figure 26 Create a new project

The application designer starts with the device window loaded with the selected device file and an empty source window.

4 Now you can start enter WML code. We will do it by using the automatic insertion help. From the **Insert** menu (see “Create or edit WML Code”) press **wml**. The header and wml tags for a wml application is written in the source window.

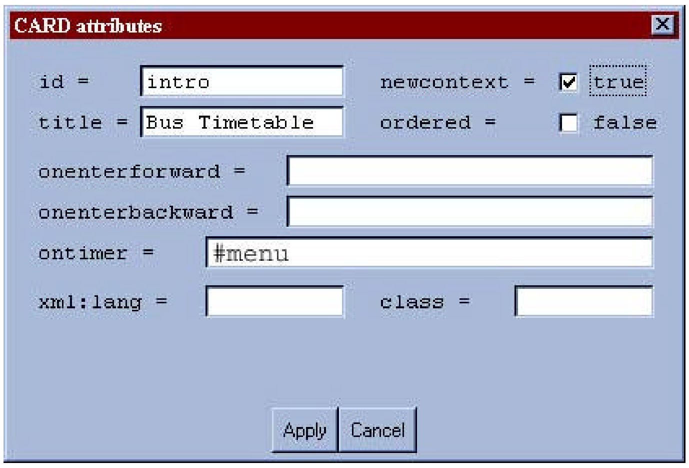


```
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.1//EN"
                     "http://www.wapforum.org/DTD/wml_1.1.xml">

<!-- WapIDE SDK WML Application Mon Jan 03 18:18:04 W. Europe Stand
<wml>
</ wml>
```

Figure 27 The first WML code

5 The cursor should now be placed at the end of the <wml>line, otherwise place it there. Press **card** in the insert menu. In the **Card attributes** window that open enter:


CARD attributes

id = intro
title = Bus Timetable
newcontext = [✓ true]
ordered = false
onenterforward =
onenterbackward =
ontimer = #menu
xml:lang =
class =

Figure 28 Insertion window for a card

6 We want an application that lets us:

- Go between cards
- Go between decks
- Have input fields
- Have select lists
- Handle variables
- Use the support for built-in timer events
- Show a picture

So enter the rest of the WML code according to the example below. If you need help in understanding the code we refer to WML guides available for example at WAP Forum, see “Related Internet sites”. Also copy the picture `C:/Program Files/Ericsson/sdk/Samples/cgi-bin/wmllib/demo/bus.gif`, to the same location that you stored your WML file, typically `C:/Program Files/Ericsson/sdk/sdk/idemain/projects/timetable/wml/`.

```
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.1//EN"
                    "http://www.wapforum.org/DTD/wml_1.1.xml">
<wml>
   <card id="intro" newcontext="true" ontimer="#menu" title="Bus
Timetable">
      <timer value="45"/>
      <do type="accept">
         <go href="#menu"/>
      </do>
      <p>
      <img width="10" hspace="1" src="bus.gif" height="20"
vspace="1" align="middle" alt="Bus Timetable"/><br/>
      </p>
      <p align="center">
         <em>Timetable Schedule</em><br/>
      </p>
   </card>
   <card id="menu" newcontext="true" title="Bus Timetable">
      <do type="prev">
         <go href="#intro"/>
      </do>
      <p>
         <strong>Choose Area </strong><br/>
         <select name="area">
```

```
          <option onpick="#line">Lund</option>
          <option onpick="#line">Stockholm</option>
       </select>
       </p>
    </card>
    <card id="line" title="Bus Timetable">
       <do type="prev">
                    <go href="#menu"/>
       </do>
       <do type="accept">
          <go href="times.wml"/>
       </do>
       <p>
          Enter line<br/>
          number:<br/>
              <input type="text" name="line_no" format="*N"
                     maxlength="3"/>
       </p>
    </card>
</wml>
```

7 In the Source window save the new WML file. In the window that appear give the file a name, for example `timetable.wml`, WapIDE will save the file by default in the folder `wml` in the project folder you created and named in step 4 above.

8 Now you are ready to test the WML code in the device in the application designer. To check that you have entered all code correctly start by compiling it. Press the "**Compile current source file**"–button in the Source window. Open the Output window if it is not already open. Check that the WML parser accepts the file (message "`no syntax errors found`").

9 Activate the device and set it in **test mode** by the "**toggle project test mode**"-button in the application designer main window. The device will start and load your timetable application from the source window. Now you can test your application by entering data and going between cards.

10 Since you have not yet created the file `times.wml` referenced in your application you will get an error that says that the file was not found. This is easy to correct. Just create a file in the same directory with the correct name and the content:

```
<?xml version="1.0"?>
<!DOCTYPE wml PUBLIC "-//WAPFORUM//DTD WML 1.1//EN"
                    "http://www.wapforum.org/DTD/wml_1.1.xml">

<wml>
    <card id="chosen" title="Your choice">
       <p>
       You chose<br/>
       <i>$area</i> and <br/>
       <i>$line_no</i>
       </p>
       <p>
          Buses leaving:<br/>
          At 14.45<br/>
          At 19.18<br/>
       </p>
    </card>
</wml>
```

11 Now its time to publish your WAP application and access it by using the browser supplied with WapIDE. (Or if you have another browser you can use that, but that is not included in this user guide.) We assume that you have a web server that is connected to the Internet and configured for WAP MIME types (see “Configure a web server”). Or that you have the Xitami web server supplied with WapIDE running. If the Xitami server was not started automatically at start up you should start it by the menu `Start/Program/Ericsson/WapIDE/WML Server`. For instructions where to place you files at another web server we refer to the instructions for that server. For the Xitami the simplest way is to use the folder
`C:/Program Files/Ericsson/sdk/Samples/wml/`
Put your files `timetable.wml`, `times.wml` and `bus.gif` there.

Note! There is no need to transfer any other files or folders since they are part of the WapIDE project structure not the application itself.

12 To access you application start the browser in WapIDE, select the R320 device and start the log window with all logs active except 'source'. The log window helps you to see what happens when you access your application.

When you start the device (by pressing the NO-button) a default page will be loaded but we want to load the new files. This is done by the menu **Load URL** in the File menu. If you followed the instructions regarding the Xitami server enter the URL according to Figure 29, and press Ok.

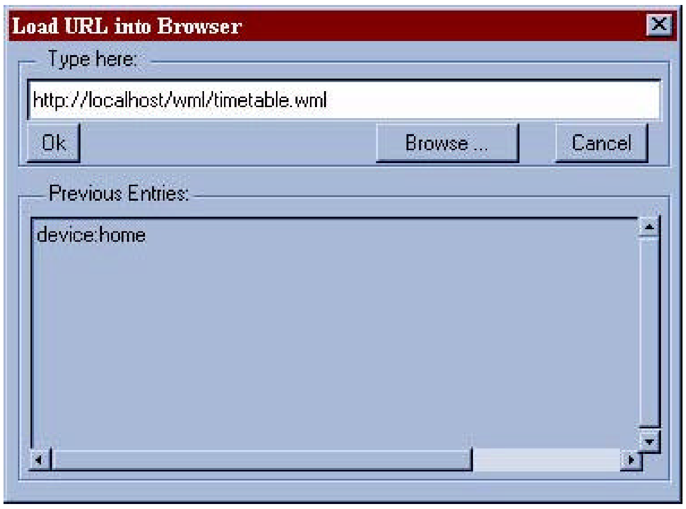
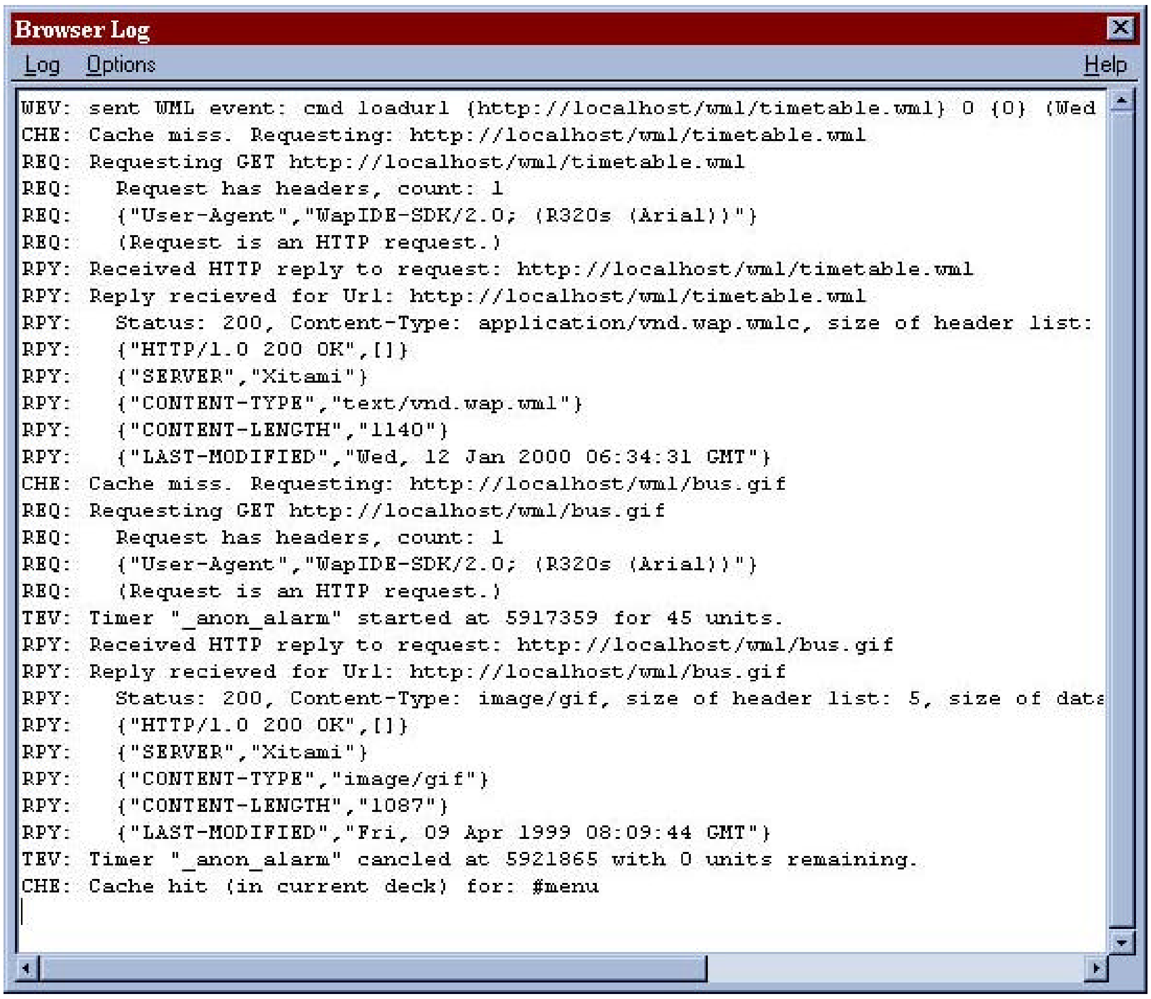


Figure 29 Enter an URL

Now the browser will load your application which means that you can test how it looks for other users it now that you have placed it at a web server.

Note! In order for all users to access you application you have to place it on a web server with access for users on the Internet.

A log from this might look like Figure 30.



```
WEV: sent WML event: cmd loadurl {http://localhost/wml/timetable.wml} 0 {0} (Wed
CHE: Cache miss. Requesting: http://localhost/wml/timetable.wml
REQ: Requesting GET http://localhost/wml/timetable.wml
REQ:   Request has headers, count: 1
REQ:   {"User-Agent","WapIDE-SDK/2.0; (R320s (Arial))"}
REQ:   (Request is an HTTP request.)
RPY: Received HTTP reply to request: http://localhost/wml/timetable.wml
RPY: Reply recieved for Url: http://localhost/wml/timetable.wml
RPY:   Status: 200, Content-Type: application/vnd.wap.wmlc, size of header list:
RPY:   {"HTTP/1.0 200 OK",[]}
RPY:   {"SERVER","Xitami"}
RPY:   {"CONTENT-TYPE","text/vnd.wap.wml"}
RPY:   {"CONTENT-LENGTH","1140"}
RPY:   {"LAST-MODIFIED","Wed, 12 Jan 2000 06:34:31 GMT"}
CHE: Cache miss. Requesting: http://localhost/wml/bus.gif
REQ: Requesting GET http://localhost/wml/bus.gif
REQ:   Request has headers, count: 1
REQ:   {"User-Agent","WapIDE-SDK/2.0; (R320s (Arial))"}
REQ:   (Request is an HTTP request.)
TEV: Timer "_anon_alarm" started at 5917359 for 45 units.
RPY: Received HTTP reply to request: http://localhost/wml/bus.gif
RPY: Reply recieved for Url: http://localhost/wml/bus.gif
RPY:   Status: 200, Content-Type: image/gif, size of header list: 5, size of data
RPY:   {"HTTP/1.0 200 OK",[]}
RPY:   {"SERVER","Xitami"}
RPY:   {"CONTENT-TYPE","image/gif"}
RPY:   {"CONTENT-LENGTH","1087"}
RPY:   {"LAST-MODIFIED","Fri, 09 Apr 1999 08:09:44 GMT"}
TEV: Timer "_anon_alarm" cancled at 5921865 with 0 units remaining.
CHE: Cache hit (in current deck) for: #menu
```

Figure 30 A log from the timetable application.

Now you have completed the steps for creating and publishing your first WML application. Congratulations!

Doing more with WapIDE

There are some extra functionality and features in WapIDE that might be good to know about.

Disassembly tool in Application Manager

You can use the application manager to disassemble encoded WML files, see "Disassembly".

Size of the application

The size of the data that is sent over the air to the client browser is an aspect that should not be forgotten. Since many WAP devices and the bearers used has limitations, it is relevant to keep the application as small as possible. If you want to find out how big an application will be when encoded to bytecode you can use the WML bytecode compiler in the server toolset, see "Use the WML bytecode compiler".

Creating a dynamic WML application

A dynamic application can mean an application whose contents (deck and card) are dynamically created and can vary depending on f.e input variables. Another meaning can be when the application calls a WMLScript which execute and return answers to the calling application.

This chapter describes how these two ways are supported in WapIDE. If you want to create a dynamic application, follow the steps described in the previous chapter and include the actions described here.

Dynamically create a static application

A dynamic way to create applications is by using scripts. Included in the WapIDE installation is a Perl library with scripts that facilitate the dynamic creation of WML decks. All functions in this library return a string of WML content. The scripts are called from WML files containing a deck of one or more cards. The scripts will execute, do their calculations, fetch HTML pages, or whatever needs to be done. They return dynamically created WML cards. An example of how to call a Perl scritp can be found in the file `sl_time.wml`:

```
<do type="accept">
       <go href="/cgi-bin/wmllib/demo/
              sl_line.pl?SLFile=$line_no"/>
</do>
```

The script called creates a WML deck for the line service. It fetches data from a formatted file and then produces cards for the user to select the direction of the route, the days and finally the times the bus stops for each stop. This means that the resulting deck that is sent to the client browser is dynamically created depending on the chosen line. The script `sl_time.pl` can be found at `C:/Program Files/Ericsson/sdk/Samples/cgi-bin/wmllib/demo/`

For more information on the Perl scripts see **Help** in WapIDE.

Create an application including WMLScript

WMLScript is component together with WML to create WAP applications. It places some procedural logic at the client side and thus reduces the need for roundtrips. Typical usage is to validate user input and access function libraries stored in the client. To use WMLScript you must first create your own WMLScript and then call it from a WML file, as in the following example.

```
<wml>
    <card>
       <p>Enter amount:<input type=“text” key=“N”/>
          <br/>
          Total = $Sum
    </p>
    <do type=“accept”>
       <go href=“calc.wmls#calcInterest($N,12)”/>
    </do>
    </card>
</wml>
```

`calc.wmls:`

```
extern function calcInterest(N,r)
{
    var Total;
    Total = Float.parseInt(N)*(r/100+1)
    WMLBrowser.setVar(“Sum”,Total);
    WMLBrowser.refresh;
}
```

Included in the Samples directory of WapIDE are examples of WMLScript and interaction with WML code. One example is the file `welcome.wml` which calls the compiled version of a script `random.wmlsc`, which then calls a static file `random.wml` to create the resulting deck for the browser.

When creating the WML application in the previous chapter you were using the application manager and the in-built support for WML creation. There is no such in-built support for creation of WMLScript but you can use the application manager as an ordinary text editor when writing the script code.

Glossary

Character Encoding	This refers to the conversion between a sequence of characters and a sequence of bytes and vice versa. Normally WML document character encoding is captured in transport header attributes such as the Content-Type's "charset" parameter, meta information placed within a document, or the XML declaration defined by XML.
Client	The device or application that initiates the request for connecting to a server.
Content Encoding	This refers to the conversion of content from one format to another. It is can also specify a particular format or encoding standard or process.
DTD	Document Type Definition
HTTP	Hyper Text Transport Protocol
MMI	Man-Machine Interface; it is the same as the UI (User Interface)
Origin Server	This is the server where a given resource resides or is created. It is usually referred to as a web server or an HTTP server.
SDK	Service Development Kit
Softopts	Soft options are actions equivalent to HTML buttons. On an Ericsson phone select the no button to get a list. Use the up and down arrows to scroll.
TCL/TK	Tool Control Language/ToolKit
UI	User Interface.
URL	Uniform Resource Locator
WapIDE	This is an acronym for Ericsson Wireless Internet's Wireless Application Protocol Integrated Development Environment.
WTA	Wireless Telephony Application. A framework for accessing the telephony related functions in a mobile terminal.
XML	Extensible Markup Language. It is one of the W3C standard languages for the Internet, and is a restricted subset of SGML.110