BAIKONUR
INTERNET/INTRANET SUITE
GETTING STARTED

CONTENTS

INTRODUCTION

Welcome to the world of ***BAIKONUR Internet/Intranet Suite***!
BAIKONUR Web server and the development tools we supply with it will enable you to quickly and easily build your own Web site with a complex functionality. The principal feature of our BAIKONUR software package is that it allows you to visually develop Web applications with the aid of Borland Delphi or C++ Builder.

Background Knowledge Required

This manual was written on the assumption that you are already familiar with the fundamental notions of Windows NT, and that Windows NT (or Windows 95) is already installed on your computer. In writing this manual we also assumed that you have a sufficiently detailed knowledge of the concept of Internet. We expect that you also know how to write programs in Borland Delphi or Borland C++ Builder. Your understanding of the workings of TCP/IP networks would also be helpful (although not necessary).
To develop applications running under the control of BAIKONUR, it would be useful to study those sections of the manual which describe the use of HTML and JavaScript. However, you do not have to read these sections to build your first Internet/Intranet information system, for the component-based visual approach to application development will enable you to start using BAIKONUR and its tools almost immediately, referring to the documentation only when absolutely necessary.
A beginner would do wise to read the Annex to this manual, where we give a brief overview of Internet/Intranet. And, of course, there is also a wealth of literature on this subject for readers of various levels of knowledge.

Where To Find Additional Information

If information contained in this manual is not enough for you, you can find additional materials on our Epsylon Technologies Web site (http://www.demo.ru). If that information is still not enough, you can contact our Epsylon Technologies technical support service (aes@demo.ru, telephones [095]-913-5608, 913-2934, 459-1333, 535-0319, 535-5349, fax [095]-913-2934, 535-5349).

You can also write to us at Epsylon Technologies, #23 Narskaya street, Moscow 125493, Russia.

What is BAIKONUR?

BAIKONURв is a software package developed by Epsylon Technologies company to help you to quickly and easily develop applications that rely on the use of Web browsers as client workstations to access databases in corporate Intranet networks, as well as via Internet. BAIKONUR proper is essentially a combination of a conventional Web server and an application server. The delivery package includes libraries for various programming environments, such Borland Delphi in C++ Builder (and C++ JBuilder in BAIKONUR's senior versions).

BAIKONUR is suitable for developing a wide variety Internet/Intranet systems for Windows NT platforms, from rudimentary to very complex. The simplest thing that BAIKONUR can do is to serve as a conventional Web server that deals with standard HTML documents. But, unlike most existing Web servers, BAIKONUR allows the user to interact with applications running on the server. The purpose of these applications can be quite arbitrary, and is limited only by the developer's imagination. One of the most common tasks that can be solved with the help of BAIKONUR is working with databases (viewing and modification of database data).

Client-Server Information Systems

Client-server architecture systems are designed so that the executable code exists concurrently both on the server and on the client. In these system, the functions of the server side are normally performed by an SQL server (marketed by such vendors as Oracle, Informix, Borland, Sybase, Microsoft, IBM, etc.), while the task of the client's workstation is to support the dialog with the user, generate requests, receive their results, and present them onscreen. There are now many well-developed tools available for rapid development of systems in such an architecture. One of the best such tools is probably Borland Delphi. The component-based approach to the development of a client workstation in Delphi appreciably speeds up the development of the system as a whole. One of the characteristic features of Delphi is that is uses visual and non-visual components. Visual components are employed for the development of interface elements, while non-visual ones are utilized to assemble the algorithmic part (including requests, processing of tables, etc.) out of ready-made components.

Unlike the file-server architecture (shared disk concept) systems, the processing of data in client-server architecture systems takes place mostly on the server side (through the use of triggers, stored procedures, etc.). The client's application merely provides access to the data on the SQL server by establishing the connection, sending requests and receiving resultant data sets. However, the fact that the user has access to the meta data that conditions the structure of database tables, etc., can be a disadvantage. Besides, it is far from simple to develop a client application that could run with equal success (if at all) in different operating system environments.

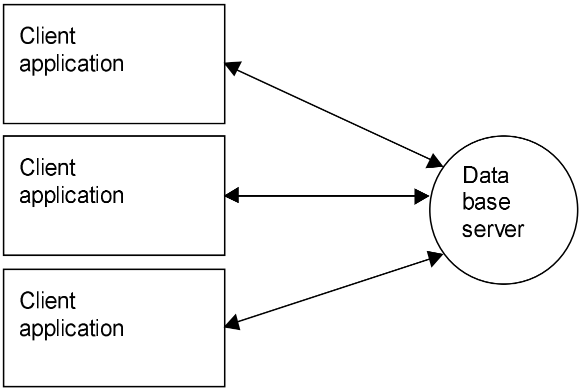


Figure 1. Classic Client-Server System Architecture

Changing To A Three-Tier Architecture

When you use BAIKONUR, an additional link (called *application server*) is introduced between the client and the server. With this approach, the applications developed with the use of rapid-development tools (such as Delphi) are executed on the server under the control of BAIKONUR, rather than on the client side. The SQL server can run either on the same computer as BAIKONUR, or on another machine in the (usually local) network.

The client workstation proper is now essentially a computer with an arbitrary selectable configuration and operating system – it can be Windows 3.11, or Macintosh, or UNIX, or OS/2, or Windows NT. Client applications access the server via the local network (using the TCP/IP protocol) or communicate with BAIKONUR server via a direct modem connection, Internet, or a dedicated communication line. Every client workstation has either a general-purpose Internet browser or a specialized browser installed on it. Using such a browser, a client establishes a connection with the Web server and launches one or several applications (visually developed in Delphi) on it, and can switch from one application to another, obtain on-line information, and perform a variety of other functions, all without loosing the connect.

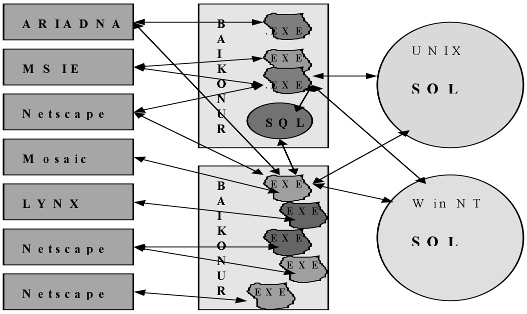


Figure 2. Three-Tier Internet/Intranet System Architecture

Advantages of a Three-Tier Architecture

Apparently, the introduction of an additional link into the client-server chain has to give the user some important advantages from the point of view of practical utilization of the resulting information system. And this is indeed so, because:

- A system that employs a BAIKONUR Web App Server is not critical to the type of the operating system used in or the computing power of client workstations. Most conventional client-server systems would not function properly unless the client computers have at least 16M of RAM. Besides, in situations when more than one type of operating systems are employed in the system, the client part of the software has to be developed individually for every one of them. In contradistinction to that, BAIKONUR makes it possible to employ standard Internet browsers on client computers, which can be minimum-RAM machines using any type of operating system.
- Client workstations can communicate with a BAIKONUR server from a remote location, and need not be appreciably modified to be able to do that. Currently, the standard client-server approach is not so far advanced as to permit a client application to run flawlessly in **remote mode** without an appropriate modification. With BAIKONUR, various frequently-asked practical questions pertaining to the use of remote workstations (like “what would happen if the line gets broken?”, “would the task context remain then?”, “is it possible to get connected to the already running tasks from another workstation?”, “how do we solve the problem of remote administration?”, “how do we cope with interference in the network lines?”, ”how do we solve the problem of access through both a modem pool and Internet?”, “how secure shall our data be?”, “what about the network traffic intensity?”, etc.) can be answered and solved with little or no difficulty, since historically Internet technologies were called upon to solve these very problems, and it is there that the greatest experience has probably been accumulated.
- A BAIKONUR-based system is *easily scaleable*, which means that you can easily upgrade it from a small system of a work group level to a huge system incorporating hundreds or even thousands of concurrent users. More than that, such expansion requires no fundamental reconstruction of the system, and can be achieved simply through increasing the power of the available application server hardware or using several application servers.
- The licensing policies applied to SQL servers and Web servers (and BAIKONUR Web App Server as such is one of them) differ greatly. An enterprise purchasing a SQL server is also obliged to buy a license for each and every workstation that will concurrently work with the server. In the case of BAIKONUR Web App Server, you have to pay only for the server license. We offer our customers different versions of BAIKONUR designed to handle different overall loads. In a situation whereby 200 users work concurrently with a database, the practical number of simultaneous connects which a BAIKONUR server establishes with a SQL server can be dozens of times as few, and this what makes it possible to **save real money** in implementing a major corporate network project. Different companies pursue different licensing policies, often giving their clients access to a SQL server through a transaction monitor or an application server, rather than directly. For example, Borland International offers a special type of license permitting an unrestricted access to Borland InterBase through a Web server.

- As soon as the number of active workstations in a client-server architecture system exceeds several dozens, the system administrator starts experiencing serious problems with the administration of such a large number of workstations. More often than not, corporate information systems are in a state of perpetual evolution, so that there is a constant need to make sure that a fresh version of client application software is installed on all the workstations, etc. When the number of workstations approaches a hundred, it becomes physically impossible to update the client application software timely. If, further, there are remote workstations in the system, its administration turns into a permanent headache for the company's technical staff. There are other things to be taken into consideration as well, including the cost of such administration at company level, unavoidable down-time hours for each workstation (such as for technical maintenance), possible crush of the entire system in a situation when incompatible versions of client application software are employed, etc. With BAIKONUR, all these problems are solved much easier. As was already said, an information system in a three-tier architecture built around BAIKONUR is not overly critical to the versions and types of Internet browsers, and will function correctly even different browser versions and different operating systems are employed. In such a system, application software has to be upgraded to newer versions only on application servers, which requires much less pain and effort than compared to the situation when they have to be re-installed on all client workstations.
- A BAIKONUR-based system is **functionally expandable**. BAIKONUR Web App Server implements a *dynamic protocol change technology*. BAIKONUR uses the HTTP as the main protocol. However, the use of the dynamic protocol change technology makes it possible to control information flows that are intended not to send and receive hypertext pages through Internet. These can be information flows designed for the transmission of voice data, e-mail messages, simply files, video information, monitoring data, measurement results, etc. With BAIKONUR, you can accomplish all this by employing an already existing standard, or develop and implement a standard of your own. This unprecedented capability to expand the system functionality is absolutely uncharacteristic for older-generation client-server software products.
- **Secrecy and safety**. It is accepted in previous-generation client-server systems that a client application has access to the system's meta data. Practical experience shows that with such an approach it is very difficult to ensure a hundred-percent security of the entire information system. Besides, only a few of the currently marketed client-server application products have the capability of an on-the-fly encryption of network data traffic.

BAIKONUR Web Application Server implements the SSL (Secure Socket Layer) standard, which means that all the information transmitted over the network is encrypted following one of the popular encryption algorithms. With the most senior version of BAIKONUR you can even change the encryption algorithm – an option that in many situations (especially when the implementation of a restricted-access corporate project is involved) is very valuable. The problems of secrecy and data safety in practical implementation of a corporate information system go far beyond the requirement that all the information transmitted over a modem line or a network cable be reliably encrypted. A BAIKONUR-based system allows you to easily keep records of every system access attempt; if necessary, you can even log any attempt to break into the system. It is also possible to write a module (an executable application) that would take immediate actions in a critical or emergency situation. Typically, every client sees a dynamically generated page in his/her browser window. This page is generated anew with every new access. In a standard case, the user has no access to the meta data or any other system information. More than that, you can make the system to generate a unique version of what every single user sees on the computer screen, in accordance with his/her access rights or some other criteria or consideration.

- **BAIKONUR permits integration of radically different systems into a single whole**. All major software vendors are trying to produce their own versions of Internet/Intranet software – a tendency that has lately become clearly apparent. BAIKONUR takes this tendency into account, and makes it possible for you to integrate the software packages of most vendors into a single system by writing missing parts in the form of executable modules run under control of the BAIKONUR server.

What You Can Do With BAIKONUR

You can use BAIKONUR to build your own Internet or Intranet Web site, and then employ it to perform typical Web site tasks, for instance:

- publish information on your company or your private information (advertising material, reports, technical information, information on your staff, etc.) on static HTML pages
- publish information about your fields of interest, either business, professional or personal
- distribute documents among members of your work group relating to the jointly implemented project(s)
- place links to the work files on your Web pages, complete with comments and annotations
- publish exotic-format information (such as multimedia information or VRML-format graphics)

Thanks to its unprecedented built-in functional capabilities as an application server, BAIKONUR allows you to implement quite complex projects by writing the server logic in Borland Delphi or Borland C++ Builder. You can create client-server systems with a multiple-tier architecture, or transfer ready-made projects written in Delphi into the Internet/Intranet environment. These capabilities allow you to master a whole new field of Internet/Intranet applications.

Thus, you can utilize BAIKONUR to:

- build a client accounting system
- create an Internet shop
- develop an Internet/Intranet merchandise system
- create a friendly, affable and smart Web site with extra intellectual features
- build a Web site that would “recognize” its visitors and generate unique pages according to their individual preferences, site visit statistics, etc.
- build a corporate information system with a complex internal logic

BAIKONUR comes with several additional utilities which you can use to solve a number of other tasks:

- *Yandex* search system to help you organize a document search and retrieval system that would take full account of the intricate morphology of the Russian language
- *Colonel* file replicator, which BAIKONUR users can employ to update files on the server
- *HTMLClient* component, intended to help you create your own clients instead of browsers

It is, of course, possible that you would require more capabilities than BAIKONUR Web App Server in its current version can offer. If that is the case, we strongly recommend that you consider upgrading to senior versions of BAIKONUR, such as BAIKONUR SuperServer.

How to Install BAIKONUR

BAIKONUR installation procedure is fairly straightforward:

- Insert the first BAIKONUR distribution diskette into your floppy-disk drive (if you install BAIKONUR from diskettes) or navigate to the directory containing BAIKONUR distribution files (if you install BAIKONUR from a hard disk or a CD-ROM)
- Run the installation program (INSTALL.EXE)
- Select the directory in which you want to have BAIKONUR installed in the dialog appearing on the screen. To select a directory other than the one offered by default, click the “Browse” button and specify the directory name (if this directory does not exist, it will be automatically created during installation)
- Click the “Next” button
- A dialog to select the components you wish to install will appear on the screen. Use this dialog to specify which of the available components should be installed: Program Files (BAIKONUR system files), VCL (visual components library to develop new applications), Samples (sample applications written in Delphi), and Yandex (search system)
- Click the “Next” button.
- A dialog offering you to select the program folder in which shortcuts to BAIKONUR files will be stored will appear on the screen
- Click the “Next” button. The installation program will copy the necessary files to your hard disk and create a program folder
- The final screen form of the installation program will be displayed
- Click the “Finish” button

This completes the installation procedure.

Before starting up BAIKONUR for the first time, please read the file README.DOC carefully.

Result of BAIKONUR Installation

Once installation is complete, the following sub-directories are created in the BAIKONUR installation directory:

SYSTEM -	System directory containing the server's executable modules and service files.
HOME -	Server's home directory containing compiled applications and the Home Page file of the server. The user will have access to the data stored in this directory and its sub-directories (besides, you can also employ the ADMIN utility to specify additional directories that will be accessible to users via aliases).
TOOLS/EXAMPLES -	Directory containing sample Delphi programs.
TOOLS/CLIENT -	Directory containing the *HTTPClient* component.
TOOLS/LIB2_01-	Directory containing modules of the HTML Controls library for Delphi 2.01.
TOOLS/LIB2_0 -	Directory containing modules of the HTML Controls library for Delphi 2.0.
TOOLS/LIB3_0 -	Directory containing modules of the HTML Controls library for Delphi 3.0.
TOOLS/LIB_BCB -	Directory containing modules of the HTML Controls library for Borland C++ Builder 1.0.
TOOLS/LIB_SRC -	Directory containing interface modules of the HTML Controls library.
ISAPI -	Directory containing the ISAPITER utility and practical examples of employing ISAPI (Internet Server Application Interface) with BAIKONUR.
CGI-BIN -	Directory containing CGI examples.
FTP -	FTP home directory (you can also specify a different directory for the FTP protocol)
DOC -	Directory containing the TXT and DOC files describing the libraries, settings, etc.
YANDEX -	Directory containing the YANDEX system that can perform document find functions with due account of the morphology of the Russian language.

Besides, a file with BAIKONUR's initial settings (BAIKONUR.INI) will be placed into Windows home directory.

Using BAIKONUR For The First Time

BAIKONUR Web App Server can function under Windows 95, Windows NT 3.51 and 4.0.

In Windows 95, you must open BAIKONUR program folder and use the "BAIKONUR for Win'95" shortcut to start up BAIKONUR.

In Windows NT, BAIKONUR can be launched as a program (in the debug mode) or as a service. In the former case you must use "Debug Mode" to start up BAIKONUR.

In the latter case, proceed as follows:

- Open the “Control Panel”.
- Select the “Services” utility.
- Select the “BAIKONUR” line
- Click the “Start” button.

If BAIKONUR refuses to run as a service, the reason can be one of the following:

- a program or a service that uses port 80 (or another port required by BAIKONUR server) has already been started earlier;
- the file BAIKONUR.INI could not be found in Windows home directory.
- The *SystemDirectory* parameter is incorrectly specified in the BAIKONUR.INI file. To correct the problem, modify the value of the *SystemDirectory* parameter in the [BAIKONUR] section of the BAIKONUR.INI file (with the help of the ADMIN utility or by directly editing the file). The system directory is the one where the file BAIKONUR.EXE and its associated DLLs are located.

The difference between BAIKONUR’s debug mode and the way it is functioning as service consists in the following.

In the debug mode, all programs are invoked for execution with the same access rights as those of the user who had started up BAIKONUR, and file access rights are determined accordingly.

When you run it as a service, BAIKONUR can identify clients by their names and passwords and assign the appropriate access rights to programs and files (depending to the name/password combination supplied by the client).

If you are a novice with BAIKONUR but want to be able to debug applications, we recommend that you run it in Windows 95 or in Windows NT debug mode.

You can start using BAIKONUR on a stand-alone computer, for example to develop an application or just study the way it works (that is, you can start BAIKONUR server and a browser on one and the same computer).

Studying Examples

BAIKONUR comes with a number of pre-compiled sample programs. After installation, they are located in the HOME directory. The purpose of these examples is demonstrate how applications written in Delphi are functioning under BAIKONUR control. For some of the samples to function properly you need to have BDE installed on your computer and specify the DBDEMOS alias.

To view the examples, proceed as follows:

- Start BAIKONUR server as described above
- Start a browser program (such as MS Internet Explorer or Netscape Navigator)
- Specify the following URL in the browser’s *Location* field: “*http://localhost/*” or “*http://<computer_name>/*”, or “*http://<computer_IP_address>/*”
- If you did everything correctly, the browser will display the home page on which there will be several links (shortcuts) to examples

- Click on one of these links with the mouse – this will launch the corresponding program which you can work with from the browser
- To close an example, click the “Close application” button

When working with sample applications, avoid using the browser’s “Back” button.

All sample applications come complete with their source texts (see the TOOLS\EXAMPLES directory). There are also source texts of several examples that are not included into the delivery package in compiled form. Before compiling them, you must have the HTML Control Components library installed in Delphi or C++ Builder as described later in this manual.

CGI Examples

Our first CGI example does no processing of user input, but allows you to view the contents of the server’s Desktop. The program is intended to be run in Windows NT 4.0 environment, and illustrates the use of the Workspace Shell capabilities in decoding shortcuts. To run the program, simply enter its name in the browser’s *Location* field:

http://localhost/cgi-bin/ws.exe

If BAIKONUR and sample applications are located on a remote computer, you must substitute the name of that computer for *localhost*.

IMPORTANT. The example *ws.exe* works with the system’s Desktop as an OLEAutomation server. Therefore DO NOT run the example when the server operates in the service mode with the Desktop interface disabled – this can crash the system.

Our second example demonstrates how forms are used and how parameters are passed to a CGI application. The example consists of the main form (REG.HTM) and the *Registr.exe* program proper. The program returns a list of parameters received from the browser via the CGI. To run the program, simply enter its name in the browser’s *Location* field:

http://localhost/cgi-bin/reg.htm

Both examples mentioned above can function under control of any Windows NT server that has a CGI support capability.

ISAPI Example

If you installed BAIKONUR with the help of the standard installation utility, you can view the result of execution of the ISAPI example by entering the following line in the browser’s *Location* field:

http://localhost/ISamples/Counter.dll

We supply a corresponding Delphi source text for this example.

FTP, Gopher And Finger Protocol Examples

Apart from HTTP, BAIKONUR Web App Server can support several other protocols, including FTP, Gopher and Finger. To be able to access the server using these protocols, you must use the corresponding client software. FTP and Gopher protocols are supported by most browsers. To try out the Finger protocol, you can use the

FINGER program supplied with Windows NT.
You can try out the FTP protocol by issuing a URL request of the following form from the browser:
ftp://localhost/
or
ftp://<computer_name>/
To try out the Gopher protocol, issue a URL request of the following form from the browser:
gopher://local host
or
gopher://<computer_name>/
You can check how the Finger protocol is functioning by running the program FINGER with parameters as follows:
FINGER@LOCALHOST
or
FINGER@< computer_name >
For example:
FINGER @www.demo.ru
For the FTP protocol, the default home directory is assumed to be *<Install_Dir>\FTP*, for example:
c:\baikonur\ftp.
For the Gopher protocol, the default index file is assumed to be *<Install_Dir>\Gopher\index.gf*, for example:
c:\baikonur\gopher\index.gf.
These default setting can be changed (see the section devoted to BAIKONUR server administration).

HTML Controls Library

BAIKONUR delivery package includes variants of HTML Controls library for Delphi versions 2.0, 2.01 and 3.0 (located in the directories TOOLS\LIB2_0, TOOLS\LIB2_01 and TOOLS\LIB3_0, respectively), plus a variant of the library for C++ Builder 1.0 (located in the directory TOOLS\LIB_BCB).

Installation Of Library For Delphi And C++ Builder

To add components of the HTML Controls library to the Components Palette of Delphi 2.0 or 2.01, do the following:

- Select the *Component|Install Component…* option in Delphi main menu
- Click the “Add” button
- Click the “Browse” button
- Navigate to the directory TOOLS\LIB2_0 (or TOOLS\LIB2_01) and select the file HTML_REG.PAS there
- Click the “OK” button
- Click the “OK” button

The library components for C++ Builder are installed similarly, except that you must select the module HTML_REG.PAS in the directory TOOLS\LIB_BCB.

Once the library is re-compiled, three new pages (“HTML”, “HTMLAdd” and “HTML DB”) will appear in Delphi’s (or C++ Builder’s) Components Palette.

The “HTML” page contains the *HTMLControl* and *HTMLPage* control elements, and visual components *HTMLLabel*, *HTMLTag*, *HTMLButton*, *HTMLImageButton*, *HTMLEdit*, *HTMLMemo*, *HTMLCheckBox*, *HTMLRadio*, *HTMLListBox*, *HTMLComboBox*, *HTMLHeader*, *HTMLRuler*, *HTMLImageRef*, and *HTMLImageRes*.

The “HTMLAdd” page contains visual components *HTMLHidden*, *HTMLList*, *HTMLTable*, *HTMLDynamicTable*, *TJavaScript* and *THTMLPutFile*, as well as four select dialog edit components (*HTMLFileListBox*. *HTMLDirListBox*, *HTMLDriveComboBox* and *HTMLFilterComboBox*).

The “HTML DB” page contains visual components employed to represent database items: *HTMLDBGrid*, *HTMLNavigator*, *HTMLDBText*, *HTMLDBEdit*, *HTMLDBMemo*, *HTMLDBCheckBox*, *HTMLDBGIF*, and *HTMLDBImage*.

The *HTMLClient* component must be installed outside the TOOLS\CLIENT directory (the HTTPCLNT.PAS module). This component is placed on a separate page.

Installation of Library for Delphi 3.0

To add components of the HTML Controls library to the Components Palette of Delphi 3.0, do the following:

- Select the *Component|Install Component…* option in Delphi main menu
- Select the option “*Into new package*” in the dialog appearing on the screen
- Specify the module HTML_REG.PAS in the “*Unit file name*” field (you can also select it on the disk by clicking the “Browse” button)
- Enter the name of the new package in the “*Package file name*” field (for example, “BAIKONUR”)
- Enter the package description in the “*Package description*” field (for example, “BAIKONUR HTML Controls”)
- Click the “OK” button
- Click “Yes” in response to the query message “*Package baikonur.dpk will be built then installed. Continue*?” appearing on the screen
- Wait until the compilation process is complete and a dialog message is displayed informing you that changes are about to be made in the package *baikonur.dpk*, and then click the “OK” button
- Once compilation is over and a message informing you of the installed components is displayed, click “OK”

Examples

BAIKONUR delivery package includes the following examples (placed in the directory TOOLS\EXAMPLES after installation):

TOTAL -	General example illustrating the use of HTML components
DBTOTAL -	General example illustrating how to work with databases and use data-aware components
NN_DEMO -	Example illustrating how an updatable screen form can be organized for Netscape browsers
LINKFORM -	Use of links to transfer to another screen form
HTMLTAG -	Example illustrating use of the HTMLTag component
JAVASCR -	Sample JavaScript application
ASKPASS -	Sample user authorization request (return of error code 401 by the HTMLControl component)
TIMELMT -	Sample program to limit the user access time
IB_CONN-	Example illustrating how to interface multiple users with InterBase (each with his/her unique name and password)
GETFISH -	Example illustrating how a client-BAIKONUR interface can be organized without the use of a browser

Compilation of Examples

You can compile the sample projects only with the HTML Controls library installed.
To compile the DBTOTAL example, the tables BIO_GIF and GUESTS should be available in the example's directory (these tables can be copied from the HOME directory).
To compile projects from the GETFISH example, you should preliminary install the HTMLControl component (located in the directory TOOLS\CLIENT).
For the DBTOTAL examples to function correctly, you must have the BDE installed on your computer.
Finally, you will need the BDE and the DBDEMOS alias for the GETFISH example, and the BDE, InterBase server and the IBLOCAL alias for the IB_CONN example.

Starting Of Sample Programs

To work with sample programs, start BAIKONUR server as described earlier in the manual. All executable programs should be located in the server's home directory or its sub-directories, or in the directories for which aliases have been specified.
The name of the home directory and the aliases are specified with the help of the server's administration utility (ADMIN.EXE).
You can call an application program for execution from a browser by specifying its URL in the browser's "*Location*" field. For instance, if the sample application program TOTAL.EXE is located in the directory C:\BAIKONUR\HOME\SAMPLES and the server's home directory is C:\BAIKONUR\HOME, this URL will have the form
http://web_serv/samples/total.exe
Note that the directory name must be followed with a slash ("/"). Do not use backslashes ("\") in your URLs.
For *web_serv*, you must substitute the following:

- actual address (for instance, localhost), if the examples and BAIKONUR are located on your computer;

- something like 194.43.88.12, if the examples and BAIKONUR are located on a remote computer and you know its IP address (194.43.88.12 in our case);
- www.some.ru, if the examples and BAIKONUR are located on a remote computer and the latter has a DNS;
- DepHost, if the computer on which BAIKONUR is installed is a local network machine.

If the program has no special termination mechanism (on a button click), it can be closed by specifying a URL of the form http://web_serv/demo.exe.! (i.e., by appending the string ".!" to the start-up URL).

The server's home directory contains the file HOME.HTM, which is used as the default home page for the server. This page contains links (shortcuts) to all the examples supplied with BAIKONUR, and is loaded into a browser when you issue a request of the form http://web_serv/.

How To Build BAIKONUR Applications

The simplest way to create a BAIKONUR application is to develop it in Delphi or C++ Builder. To assist you, we have included the HTML Controls library (separate versions of it for Delphi 2.0, 2.01 and 3.0, and for C++ Builder 1.0) in our delivery package. The way this library is installed is described earlier in this manual.

It is, of course, possible to develop BAIKONUR applications with the help of other programming tools (such as Borland C++), but you will have to master BAIKONUR API to be able to do that.

In this manual, we shall discuss the simplest approach – the building of BAIKONUR applications in Delphi.

Simplest Demo Example

1. Write a new project in Delphi (or C++ Builder).
2. Place the *HTMLControl*, *HTMLPage*, *HTMLHeader*, *HTMLRuler* and *HTMLButton* components from the "HTML" page of the Components Palette on the form. Your form will appear approximately like shown in Figure 3:

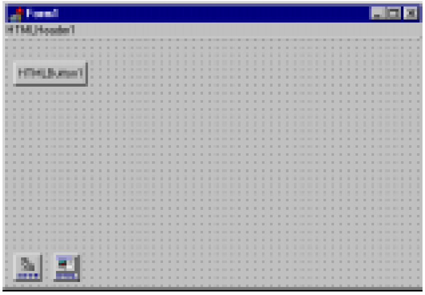


Figure 3. Initial View of Project Form at Design Time

3. Set the components' properties as indicated in the following table:

Component.Property	Value
HTMLPage1.Title	*'Simple Demo'*
HTMLPage1.HideForm	True
HTMLHeader1.Caption	'Simple Demo'
HTMLButton1.Caption	*'Close Application'*

4. Write the *OnClick* event handler for the *HTMLButton1* component (for example, by double-clicking on the component at design time) and add the '*close application*' code (add the line marked with the asterisk {*}):

```
Procedure TForm1.HTMLButton1Click (Sender: TObject);
begin
  {*} HTMLControl1.UserClose;
end;
```

5. Your project is ready now. Save it to disk. For the sake of simplicity, you can save the project using its default name in a sub-directory of the BAIKONUR's home directory. The final view of the project at design time is illustrated in Figure 4.

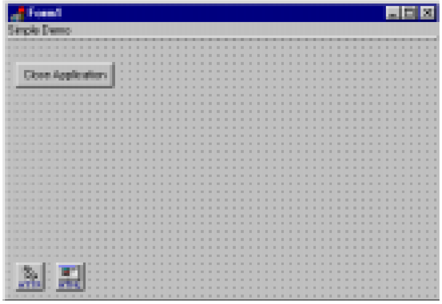


Figure 4. Final View of Project Form at Design Time

6. Compile the project (after which you may close Delphi).
7. Start BAIKONUR server and a browser (such as AMSD Ariadna). Specify a URL of the following form in the browser: http://localhost/examples/project1.exe. This will start your application, and the browser will display the form you have just designed:

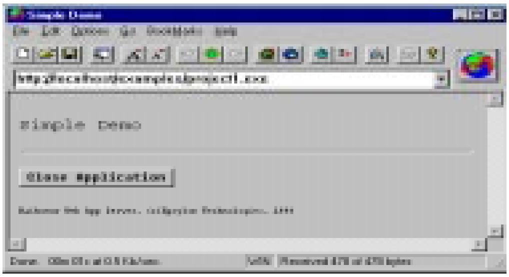
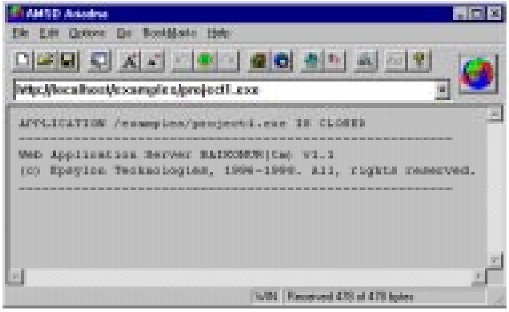


Figure 5. View of Form in *Ariadna* Browser

This is a simplest demo application. When its “Close Application” button is clicked, the application will be closed and the browser will display the following information:



```
APPLICATION /examples/project1.exe IS CLOSED
------------------------------------------------------------
Web Application Server BAIKONUR(tm) v1.1
(c) Epsylon Technologies, 1998-1999. All, rights reserved.
------------------------------------------------------------
```

Figure 6. “Application Closed” Screen

Application for Working with Databases

You can use the components from the “Data Access” page to access database data from an application, and the data-aware components from the “HTML DB” page to have the data displayed onscreen.

1. Write a new project in Delphi (or C++ Builder).
2. Place the *Table* and *DataSource* components from the “Data Access” page, the *HTMLControl*, *HTMLPage* and *HTMLButton* components from the “HTML” page, and the *HTMLNavigator* and *HTMLDBGrid* components from the “HTML DB” page of the Components Palette on the form. Your form will look approximately as shown in Figure 7:

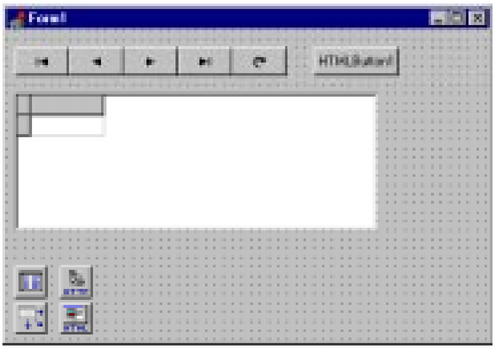


Figure 7. Initial View of Project Form at Design Time

3. Set the components’ properties as indicated in the following table:

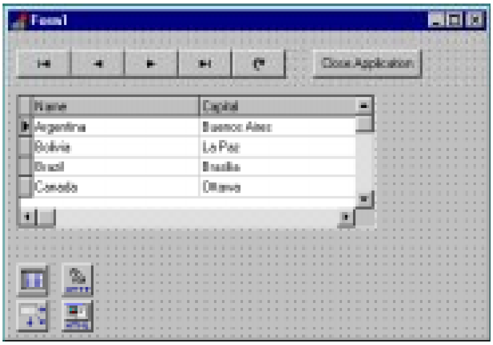
Component.Property	Value
HTMLPage1.Title	*‘DataBase Demo’*
HTMLPage1.HideForm	True
Table1.DatabaseName	DBDEMOS

Table1.TableName	COUNTRY.DB
Table1.Active	True
DataSource1.DataSet	Table1
HTMLDBGrid1.DataSource	DataSource1
HTMLNavigator1.DataSource	DataSource1
HTMLButton1.Caption	*‘Close Application’*

4. Write the *OnClick* event handler for the *HTMLButton1* component (for example, by double-clicking on the component at design time) and add the ‘*close application*’ code (add the line marked with the asterisk {*}):

```
Procedure TForm1.HTMLButton1Click (Sender: TObject);
begin
  {*}  HTMLControl1.UserClose;
end;
```

5. Your project is now ready. Save it to disk. For the sake of simplicity, you can save the project using default names in a sub-directory of BAIKONUR’s home directory (for instance, in HOME\EXAMPLE). Suppose you name your project DBAPP, and give the module the name DBAPP_U1. The final view of the project form is shown in Figure 8.



Name	Capital
Argentina	Buenos Aires
Bolivia	La Paz
Brazil	Brasilia
Canada	Ottawa

Figure 8. Final View of Project Form

6. Compile the project.
7. Start BAIKONUR server and a browser. Specify a URL of the following form in the browser: http://localhost/examples/dbapp.exe. This will start your application, and the browser will display the form you have just designed:

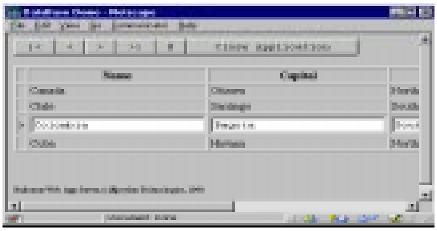


Figure 9. Form View in Netscape Navigator

Using the form's navigator buttons, you can navigate through the table and modify (edit) its data.

More detailed information on the techniques employed to develop BAIKONUR applications can be found in "Programmer's Guide".

BAIKONUR Administration Basics

The settings and parameters affecting the way BAIKONUR server is functioning are stored in the following system files:

- BAIKONUR.INI file. After installation, this file is located in Windows home directory (for example, in C:\WIN95)
- Aliases file. This file (named ALIAS.FIL by default) is located in BAIKONUR's system directory (for example, in D:\BAIKONUR\SYSTEM)
- Clients file. This file (its default name is CLIENTS.FIL) is likewise located in BAIKONUR's system directory (for example, in D:\BAIKONUR\SYSTEM)

You can set and modify the server parameters with the aid of the administration utility ADMIN.EXE, or by directly editing the above-mentioned system files.

In this section, we shall discuss the use of the ADMIN utility. For a detailed description of the format of the system files and their parameters refer to the "BAIKONUR Server Administration" guide.

The parameters of BAIKONUR server can be divided into the following groups:

- System parameters, including the settings for the supported protocols, log settings, etc.
- Settings of the server's sections (virtual servers)
- Aliases for the server's directories and names of its sections
- Additional client names

The ADMIN utility can also used (depending on the delivery package variant) to specify the parameters of BAIKONUR's auxiliary tools (such as the Yandex search system and the file replication utility).

General

Sections (Virtual Servers)

Depending on its version, BAIKONUR Web App Server can support several Internet data transfer protocols: HTTP, FTP, Finger and Gopher. The list of supported protocols can be expanded, because you can connect new protocols to BAIKONUR server. Each of these protocols can be associated with one or several TCP/IP port(s), although the "one port – one protocol" rule should be invariably observed.

One and the same BAIKONUR server can be accessed with the use of different protocols and through different ports. Besides, one and the same Web (or FTP, or Gopher) server can be made known in Internet under different names. Moreover, it is even possible to access the server on a separate computer using different names, including the computer's name, its *localhost* name or IP address (for example,

127.0.0.1).
By assigning appropriate server names and port numbers, you can define individual sections for the BAIKONUR server (otherwise known as “virtual servers”), and then associate each of them with a specific mix of aliases and system users, define the virtual servers’ reaction to certain file requests, etc.
For example, you can define a certain section (named PUBLIC, for instance) for port 80 and a server with the name www.some_web.ru that uses the HTTP protocol, and specify a home directory for that section (say, D:\BAIK\HOME). Once you do that, all users accessing the server using a URL of the form http://www.some_web.ru/ … will be receiving data from the directory D:\BAIK\HOME. Or you can create a section (named PRIVATE, for example) for port 8080 and a server with the name Corporate_Web, and specify that the home directory of that section is, say, D:\WEB\HOME. Then, the users accessing the server over the local network using a URL of the form http://Corporate_Web:8080/*…* will be receiving data from the directory D:\WEB\HOME.
The settings of a specific section override the server’s default settings.

Aliases

The data and programs accessible to BAIKONUR server’s clients are stored in its home directory. However, you can specify a unique directory for each section (virtual server). Additionally, you can specify aliases for directories and names for each individual section or the entire server.
Suppose you defined the home directory of server www.some_web.ru as C:\BAIKONUR\HOME. Then, every client accessing this server by issuing a URL of the form http://www.some_web.ru/index.html will receive the file C:\BAIKONUR\HOME\INDEX.HTML (if there is one). If you define an alias (say, PUB) for the directory D:\PUBLIC\HTML, the client accessing the server by issuing a URL of the form http://www.some_web.ru/pub/index.html will receive the file D:\PUBLIC\HTML\INDEX.HTML (again, if there is one).
This is how aliases for directories work.
But you can specify an alias for a name as well. However, a name alias should begin the an asterisk (*). If, for example, you define an alias with the name *SEARCH.HTM=SEARCH.EXE, the client issuing a URL of the form http://www.some_web.ru/search.htm will think that he/she is working with HTML file SEARCH.HTM, whereas actually the program SEARCH.EXE will be invoked for execution.

System Users

Clients accessing BAIKONUR server can receive files from and run programs on the server. In many situations it is rather important to determine what files and programs a given client has the right to access and run, as well as what rights the client-initiated programs have when executing on the server. The way the rights of the clients accessing BAIKONUR server are determined depends on the operating system used and the server’s mode of operation, and is accomplished as follows:

- When BAIKONUR is running under Windows 95, no check of user access rights is performed

- When BAIKONUR is running under Windows NT in the debug mode, the external client and his/her program are granted the rights of the user who had started up BAIKONUR
- When BAIKONUR is running under Windows NT as a service and the rights verification option is disabled, the client (and the programs he/she invokes for execution) are granted the rights of the service
- When BAIKONUR is running under Windows NT as a service and the rights verification option is enabled, the client (and the programs he/she invokes for execution) are granted the rights defined in the system clients file (normally, the file CLIENTS.FIL)

The clients file contains specifications of the external user names and passwords, as well as their corresponding Windows NT system names and passwords. If the client specifies an external name and a password matching those defined in the clients file, he/she will be granted the corresponding rights. Otherwise, his/her rights will be limited to those of an Anonymous user.

Apart from the clients defined in the system clients file, two special kinds of users (Administrator and Anonymous) are defined for a BAIKONUR server (or rather for each of its sections). You can specify in the server settings what users defined in Windows NT correspond to such users. If the user trying to log on to the server specifies a name and a password matching those assigned to Administrator, he/she will be allowed to perform some additional functions (for instance, shut down the server). Otherwise (i.e., if the user logs on without specifying a name and a password or supplies a name and a password that do not exist in the clients file), the rights of all users will be limited to those of an Anonymous user.

ADMIN Utility (Baikonur)

The ADMIN utility can be invoked for execution after installation from BAIKONUR's program group. The program file ADMIN.EXE should be located in BAIKONUR's system directory. The ADMIN utility's dialog form consists of two panes (ref. Figure 10). The left-hand pane displays a tree of parameter groups. When a specific group is selected, the right-hand pane displays a set of parameters of that group, which the user can then view and edit.

Let us briefly discuss the groups parameters (in the order they appear in the groups tree).

General Settings

You can use the server's general settings dialog (ref. Figure 10) to specify BAIKONUR's system directory, its home directory, and default page.

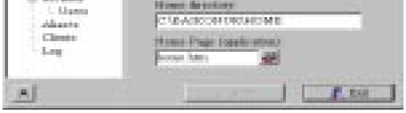


Figure 10. General Settings Dialog

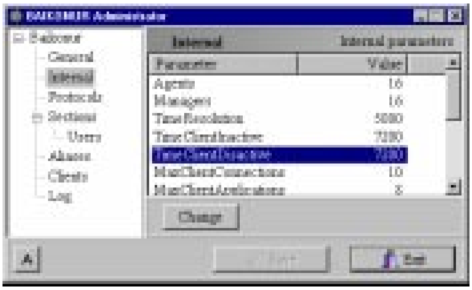
The “System directory” input filed is used to specify the “default” path of all subsequent file declarations in your Baikonur server’s INI file (the aliases file and the LOG file). A system directory is the one in which the server’s executable file BIAKONUR.EXE and its accompanying DLLs are located.

The “Home directory” input filed is used to specify the home directory for the server. It is this directory (and its sub-directories) that the clients will have the right to access (unless you also specify aliases for other directories and/or files). Note that the home directory can also be re-defined in the aliases file for the default section (for example, as “/=c:\pub\www”), in which case the alias will have a higher priority.

The “Default page (application)” input field is used to specify the page to be sent out or the application to be run when a client accesses the server without specifying a particular URL. If, for example, the default page is defined as HOME.HTM and the client accesses the server by issuing a URL like http://web_serv/ from a browser, he/she will be sent the HTML page HOME.HTM.

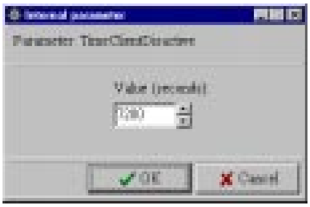
System Parameters

This page is employed to adjust some internal parameters of BAIKONUR server that determine the way it will be functioning (ref. Figure 11). In most cases, there is no need to alter the default system settings, with a possible exception of such parameters as *Applications* (maximum total number of server-started applications) and *MaxClientApplications* (maximum number of user-started programs).



Parameter	Value
Agents	16
Managers	16
TimeResolution	5000
TimeClientInactive	7200
TimeClientInactive	7200
MaxClientConnections	10
MaxClientApplications	8

Figure 11. System Settings Dialog



Parameter: TimeClientInactive

Value (seconds)

7200

Figure 12. Parameter Value Edit Dialog

If you wish to alter the value(s) of a system parameter, click the “Change” button. This will bring the dialog form shown in Figure 12 to the screen.

Enter a new value and click “OK”. To cancel the operation, click the “Cancel” button.

Valid values of the system parameters are given in the file BAIK_INI.DOC, as well as in the “BAIKONUR Server Administration Guide”.

Setting Up Protocols

This dialog allows you to view and edit the list of supported protocols and specify the corresponding ports to be used, libraries, protocol parser functions, etc. For example, you can assign more than one port for one and the same protocol. New protocols to be supported by BAIKONUR are developed in the form of Dynamic Link Libraries (DLLs), and are added to the system using this dialog.
The protocols set-up dialog form is illustrated in Figure 13.

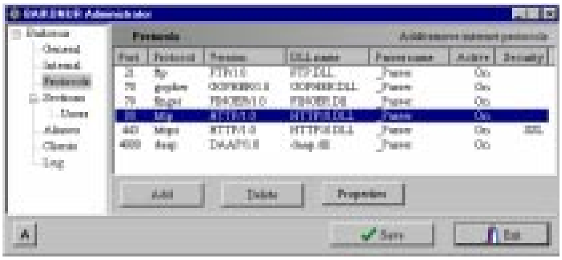


Figure 13. Protocols Set-Up Dialog

To add a new protocol, click the "Add" button. This will bring the dialog form shown in Figure 14 to the screen.

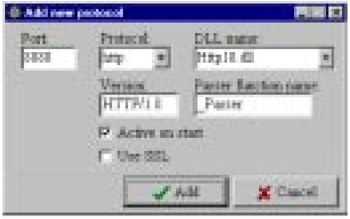


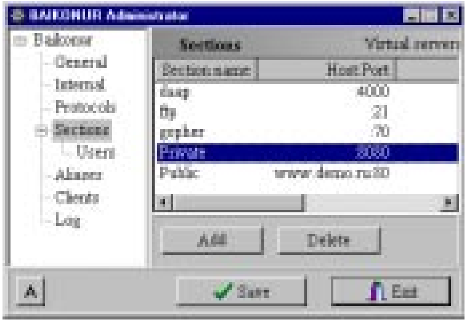
Figure 14. Adding a New Protocol

The "Port" option is used to specify the port number for the protocol. Note that some protocols have certain port numbers reserved for them. Thus, the FTP protocol uses port 21 by default, while the HTTP protocol works is assigned to port 80.
The "Protocol" option is used to specify the name of the protocol (HTTP, FTP, etc.). You can select the protocol name from a drop-down list or enter it in the input field.
The "Version" option is employed to specify the protocol's version number.
The "Protocol's DLL" option is used to specify the name of the DLL for the given protocol. The DLL must be located in BAIKONUR server's system directory, and can be selected from a drop-down list. BAIKONUR will then load the selected DLL and invoke the necessary protocol parser from there when data is received through the selected port.
The "Parser function name" option is used specify the name of the protocol parser function. For protocols supplied with BAIKONUR, the name of this function is *_Parser*.
The "Enable SSL" option allows you to determine whether or not the data passing through the selected port will be encrypted using the SSL protocol. This parameter is valid only for the HTTP protocol.
To add a new protocol, click the "Add" button. To cancel the operation, click the

"Cancel" button.
To delete a protocol from the list of protocols, click the "Remove" button on the main dialog form.

Sections (Virtual Servers)

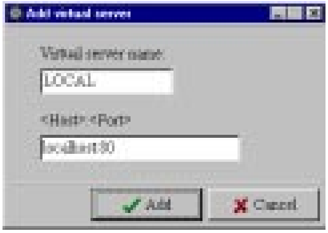
The dialog form shown in Figure 15 can be used to make a list of the server's sections, or virtual servers (discussed earlier in this manual). For each section you can specify system users, aliases and server's response to certain types of resource requests.



Section name	Host:Port
daap	4000
ftp	:21
gopher	:70
Private	8080
Public	www.demo.ru:80

Figure 15. Setting Up Sections (Virtual Sections)

A section definition consists of a unique identifier, port number and server address. The identifier is essentially a sting of up to 16 characters. The server address is a numeric IP address or DNS (such as 177.77.7.98 or www.demo.ru). For example, you can assign the name PUBLIC for the server www.demo.ru) and port 80, and the name PRIVATE for port 8080.
To add a new section, click the "Add" button. This will take you to the dialog form shown in Figure 16.



Virtual server name: LOCAL
<Host>:<Port>
localhost:80

Figure 16. Adding a New Section

The "Section name" field is used to specify a unique name for the section (it can be up to 16 characters long and should include no spaces or commas).
The "<Host>:<Port>" field is used to specify the server address followed, after a colon, with the port number. Examples: www.demo.ru:80, localhost:8080. You can have several Internet addresses reserved for one and the same server (so that the server would respond differently when it is accessed by different names).
If you do not specify the server's symbolic address in the section definition, the

settings for that section will be effective for all the ports. More information on section definition techniques can be found in the file BAIK_INI.DOC and the “BAIKONUR Server Administration” guide.

To add a new section to the list, click the “Add” button. To cancel the operation, click the “Cancel” button.

To delete a section from the list, click the “Remove” button.

Section System Users

IMPORTANT: *The following settings have no meaning unless BAIKONUR is running as a service with the rights authentication option enabled.*

As was already mentioned above, BAIKONUR allows you to define remote clients’ access rights to file read and program startup functions for each of the server’s sections and for the server as a whole (the “default” section). This is done by introducing the notions of *Administrator* and *Anonymous* users for every section.

Administrator is the user allowed to perform certain additional operations with a BAIKONUR server (for instance, shut it down).

Anonymous is the user who logged on to the server without specifying a name and/or password (or specified a name and/or password not registered in the server).

All users registered in Windows NT are made to correspond to either *Administrator* or *Anonymous*. It is important to realize that an *Administrator* for a BAIKONUR server is not the same as the administrator for Windows NT– it can be a conventional user.

The dialog form employed to set up system users for a server section is illustrated in Figure 17.

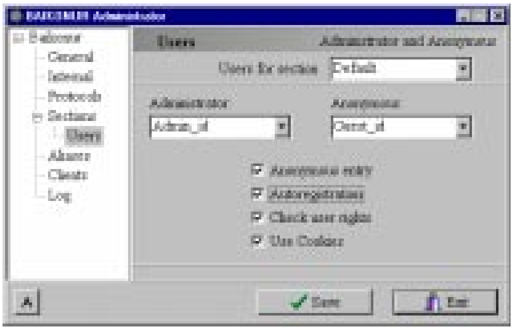


Figure 17. Setting Up of Section Parameters

The name of the section is selected from the “Users for section” drop-down list.

The users to be treated as *Administrator* and *Anonymous* are selected from, respectively, the “Administrator” and “Client (Anonymous)” lists, which contain the clients’ identifiers from the clients file (we shall discuss this later in the text).

For every section you can specify some additional options, including:

“Enable logon as Anonymous”. If this option is left unchecked, the user will have to enter his/her name and password to be able to log on. If the user-supplied identification data does not match that stored in the clients file, the logon will be denied. If this option is checked, the user will not have to supply his/her name and password to log

on.
“Enable AutoRegistration”. If the logon as Anonymous is disabled, this option determined whether the new user will be registered using his/her name and password (if these are not already available in the clients file). If this option is checked, the user may supply any name and password, and will be allowed to continue working.
“User rights authentication”. If this option is checked, the user access rights will be verified against the clients file information (i.e., the user will be granted the rights of one of the NT users). If this option is left unchecked, no authentication is performed, and the user (and his/her tasks) will have the same rights as those of the BAIKONUR service.
“Use Cookies”. The Cookies mechanism helps identify clients even if they are working through a proxy server (i.e., have identical IP addresses). This capability proves especially important in situations when a client works with an application. However, some browsers (especially their older versions) do not support this mechanism. You can disable the Cookies support if you think it necessary for some reason.

Setting Up of Aliases

Figure 18 shows the dialog form employed to set up aliases for the server and its individual sections.

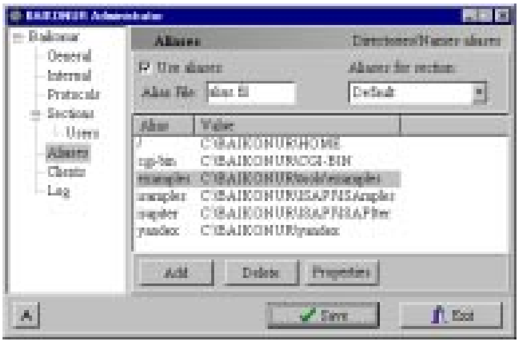
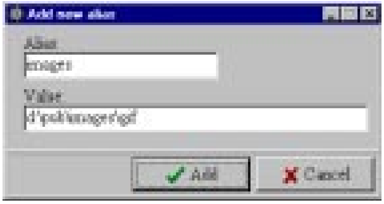


Figure 18. Setting Up of Section Aliases

The “Use aliases” option determines whether or not the server is to take aliases into consideration. We recommend that you do not disable this option unless absolutely necessary.
The “File” option is used to specify the name of the aliases file (without a path to it). The aliases file (its default name is ALIAS.FIL) is located in the BAIKONUR’s system directory.
The “Aliases for section” option allows you to select the section for which you want to have aliases defined.
To define a new alias in the selected section, click the “Add” button. This will take you to the dialog form shown in Figure 19.



Alias: images

Value: d:\pub\images\gif

Figure 19. Adding a New Alias

Enter the name of the alias in the form's "Alias" input field.
Enter the hard disk directory corresponding to the alias in the "Value" input field.
Click the "Add" button to add a new alias to the list, or the "Cancel" button to cancel the operation.
The aliases defined for the server's default section (i.e., aliases for the entire server and all protocols) have no effect upon the set of aliases defined for other sections. This means that if you define a section, the default aliases will be "invisible" when this section is accessed.

Clients File

In this file, each of the registered external names and passwords is made to correspond to a certain Windows NT user. The dialog form in which the appropriate information from the clients file is displayed is illustrated in Figure 20.

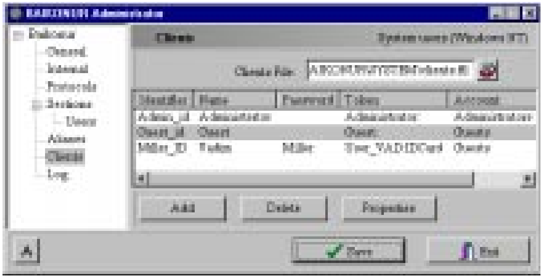
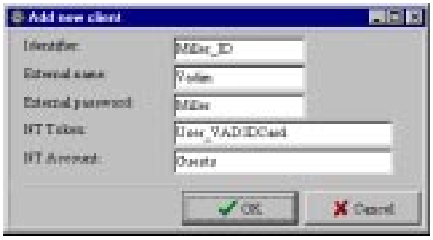


Figure 20. BAIKONUR Server Clients

The name of the file containing information on the server's clients is displayed in the "Clients file" input field, and can be altered.

IMPORTANT. *In this version of BAIKONUR, the clients file is a conventional INI file. Therefore it is important to make sure that this file is not accessible to external clients.*

Every client is assigned an identifier, an external name and a password (matching those he/she had supplied in the browser authorization dialog), his/her name and password as an NT user, and certain other parameters.
To define new user press "Add" button, to change user's properties press "Properties" button.



Identifier: Miller_ID
External name: Vadim
External password: Miller
NT Token: User_YAD:IDCard
NT Account: Guests

Fig.21 New user dialog.

The "Identifier" input field is used to specify a unique user ID string. The string can be up to 16 characters long, and should include no blank characters or commas.

The "External name" and "External password" input fields are used to specify the name and the password, respectively, which the client is required supply in the browser's authorization dialog (normally, during his/her first access to the BAIKONUR server).

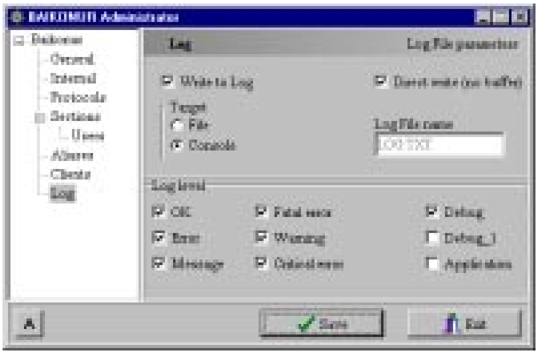
The "NT Token" input field is used to specify the name and password (separated by a colon) assigned to the client in Windows NT system.

The "NT Account" input field is used to specify the user group whose rights the client is granted when working with files.

To save the information, click the "OK" button. To cancel the operation, click the "Cancel" button.

LOG File Settings

BAIKONUR server makes it possible to keep track of messages and have them output either to the computer console or to a special LOG file. What messages are to be output and where they are to be output is determined by the settings of the LOG file set up dialog form shown in Figure 22.



☑ Write to Log
☑ Direct write (no buffer)

Target
○ File
● Console

Log File name: LOG.TXT

Log level
☑ OK ☑ Fatal error ☑ Debug
☑ Error ☑ Warning ☐ Debug_1
☑ Message ☑ Critical error ☐ Application

Figure 22. LOG File Settings

The "Keep Log" option enables/disables the message logging function.

The "Write directly to Log" enables/disables the message buffering function.

The "Output" option determines where messages are to be output ("To file" or "To console").

The “Log file name” option determines the name of the file in which messages are to be logged.
The “Save to Log” option determines what kinds of messages are to be output:

- “OK” - an “operation successfully executed” message
- “Fatal error” – a fatal error warning message; this message is displayed onscreen as a STOP dialog box regardless of the Log settings
- “Message” – messages output by the client into the server’s Log
- “Error” – other error messages
- “Warning” – an error or access rights violation warning message
- “Critical error” – a critical error warning message (i.e., error that may lead to a system crush)
- “Debugging” and “Debugging_1” – debugging messages
- “Application” – messages output by an application into the server’s Log