DOWNLOAD PDF THE NETSCAPE LIVEWIRE SOURCEBOOK

Chapter 1 : Ted Coombs | Open Library

The Netscape LiveWire sourcebook. by Ted Coombs. Publication date Topics Netscape., World Wide Web., Java (Computer program language) Publisher J. Wiley.

He was an early author of the " He is a futurist for the web site Futuristyx formerly known as Future News Network and information security expert at Risk Mitigate. One evening, while practicing roller skating , he had the idea that he would like to roller skate across the United States to protest the energy crisis that was causing long gasoline lines and gas rationing in United Artists saw this as a way to promote the movie and sponsored his venture. He later learned that Clinton Shaw had already roller skated across both Canada and the United States breaking world records. He decided to skate further, returning across the United States. He left in May from a Hollywood gas station, [2] converted to appear like a restaurant attended by stars from the movie, including Harvey Korman, to begin his trip across the US and back. He was followed by a van driven by his friend and musical partner, Brian Douglas. Along the way he broke the record for miles skated in one day by skating miles He headed North and was given a parade in Chicago followed by Playboy Bunnies skating in costume to greet the mayor of Chicago, Jane Byrne. The biggest event on his trip was entering New York City. Accompanied by a police escort he was the first person ever to roller skate through the Lincoln Tunnel. He was the guest of honor at a Central Park concert given by Eddie Money and later attended the premier of the movie Americathon. His trip was not complete. While he had fulfilled his agreement with United Artists to make it to New York, his goal was the distance record. He headed south to Richmond, Virginia and then west into Missouri , where on September 13, he broke the World distance record in the small town of Mt. That night, the driver of the van, someone other than Douglas who had left earlier in the trip stole the van, abandoning Coombs in Mt. Ted continued skating into Sedalia, Missouri where roller skate rink owners in Missouri and Leavenworth, Kansas , had skate-a-thons and raised money for his continued trip. Coombs continued until Yates Center, Kansas where he stopped his trip, [3] having skated 5, miles 8, He took a bus to Las Vegas in order to retrieve the van and then returned to his home in Hermosa Beach. Ted Coombs; Jason Coombs March Ted Coombs; Jason Coombs Jason Coombs; Ted Coombs March PowerBuilder 4 programming for dummies. Create and Manage a Java-based Web site. Jason Coombs; Ted Coombs March 23, Setting up an Internet site for dummies.

Chapter 2 : NetObjects Fusion Documentation - Database Publishing

The Netscape LiveWire Sourcebook is a hands-on guide to efficiently creating and managing interactive multimedia content on your Web site or company intranet. This book will teach you the features and capabilities of the complete set of Netscape Web development tools and technologies, including.

Enabling advanced server-side solutions is what LiveWire is all about, and if you need a complete web-to-database connectivity and management solution, LiveWire may be just what you are looking for. LiveWire is an all-in-one web site management and application development tool suite from Netscape. Sure, the LiveWire tool suite with its WYSIWYG web-site manager, web-page editor, and application manager is great for creating and managing web sites and intranets, but as a database developer, you want to know how you can use LiveWire for web-to-database solutions. This is where the LiveWire server extensions come into the picture. Using the server extensions, you can build powerful database management applications that run within the context of a web browser. Your Netscape client can be any version of the Netscape Navigator browser, including Netscape Navigator 3. Netscape recommends that you use LiveWire with its SuiteSpot servers. Still, as long as you have version 2. Currently, LiveWire is available in two editions: The LiveWire professional edition is included in the SuiteSpot package. Ideally, you will install LiveWire on a development machine and deploy your finished and tested LiveWire applications to your web server. However, keep in mind that to run applications after you build them, the development machine must be running a Netscape server process. Site Manager is the main interface to everything LiveWire has to offer. You will use Site Manager to generate web sites based on templates, to manage your web site or intranet, and to compile your LiveWire applications. The server-side JavaScript objects provide an extensive framework for working with databases, accessing files, and tracking the state of clients, applications, and servers. After you enter all the necessary HTML markup and JavaScript statements, you use the build feature of Site Manager to create server-ready files for the application. The Application Manager also features access control and debugging mechanisms. After you build an application, you must add it to the manager, then you must start the application, which tells the server the application is available to users. To create a LiveWire application, you start with source files that are displayed in a web browser as web pages. The hypertext linking and redirection capabilities of the web mean that LiveWire applications can have any number of source files associated with them. It is the structure of these files that defines the interface for your application. Although LiveWire applications are preparsed, your web server will always parse the data again before sending it to a client. As stated earlier, LiveWire is designed to be used with Netscape clients. JScript and JavaScript are not percent compatible, which means that some LiveWire applications will not work properly in Internet Explorer. Keep in mind that Microsoft and Netscape are discussing a standardized version of JavaScript that will bring the features of JScript and JavaScript into a single mainstream specification. The LiveWire Object Framework When you install LiveWire, the server extensions and server-side object framework are installed on your web server. The server extensions are responsible for translating your server-ready files that contain LiveWire object references into usable content. The LiveWire object framework consists of the server-side objects listed in Table 1. The most important object for database transactions and management is the database object. The database Object LiveWire is the intermediary between client browsers, your web server, and the database server. When a client browser initiates a database transaction, your LiveWire application which is running on the web server receives the request, processes it into a form acceptable to the database server, then makes a request to the database server. After processing the request, the database server returns a response to the LiveWire application and, in turn, the application formats the response and sends it to the client. The object that makes this interaction possible is the database object. As with most JavaScript objects, the database object has methods associated with it. Methods used to connect to a database server and report errors are shown in Table 2. Methods used to handle queries, transactions, and cursors are listed in Table 3. LiveWire uses the methods of the database object to create generic instructions that can be passed to any ODBC-compliant database. Before the instructions are passed to the database server, LiveWire translates them into a series of

SQL statements. Data Type Conversion Although most databases support many different data types, JavaScript and LiveWire support only a limited number of data types. As a result, LiveWire must convert incoming data from your database to data types supported by JavaScript, and must convert outgoing data to data types supported by your database server.

Chapter 3 : unsubclass - Wiktionary

Note: Citations are based on reference standards. However, formatting rules can vary widely between applications and fields of interest or study. The specific requirements or preferences of your reviewing publisher, classroom teacher, institution or organization should be applied.

Working closely with Netscape, leading Internet product and service partners developed these reusable business applications for Netscape SuiteSpot. Our goal is to give you the tools to create robust JavaScript and Java apps and enable you to quickly prototype an intranet based on the Netscape ONE platform. The majority were developed using Netscape LiveWire Pro and make extensive use of JavaScript , with some incorporating Java applet functionality. Most of the apps are database-driven, which offers several benefits to intranet and Internet sites: For example, when employee data is collected for a benefits enrollment application, the contact information also can be used for an employee phone directory. The source code you download provides a foundation for you to create complete applications. How can I use the intranet tools featured in this program? Each tool description page has links to the download pages. Do I need any specific software to use the quick start applications? Please see our download area for full information on the required software and system requirements. A Web server that supports JavaScript. To build applications, you need the LiveWire Compiler, which is bundled with Enterprise. With FastTrack, you need to purchase LiveWire separately to get the compiler. Which platforms are supported? On the client side, because Java and JavaScript are open languages, the applications can be accessed across platforms from clients that support Java. The applications are designed to perform optimally with Netscape Navigator 3. Which databases are supported? The code for these applications is written with an Informix database schema. However, you can modify the code to work with Oracle, Sybase, and dozens of additional databases, from desktops to mainframes. By changing the database connect statement and making minor changes to embedded SQL, developers can take advantage of the applications with any of the databases supported by Netscape LiveWire. JavaScript calls to the database are both database and platform independent. Thus you will be able to access data from your existing databases, or develop new databases and data warehouses. Are these applications secure enough to use for confidential information? Your data security depends largely on the login protection you design into your modified apps, and where the data resides. How reliable are these applications? Netscape has tested these sample applications for features and functionality. Because they are designed as "templates" for developers to use as the basis for developing complete applications, we and our partners have not performed the exhaustive quality assurance checks typical for shipping applications. Thus, we do not recommend using the applications "as is" for mission-critical purposes. Where can I find more information on a particular application? How do I get support? In general, neither Netscape nor our partners are providing support for these free applications. However, we offer discussion forums for each application and tool, as well as several related to Netscape. These forums should be a valuable source of information and advice from the vendors monitoring the forums as well as IS professionals using the software to build intranets. In fact, many of these apps could serve as templates for complete applications you develop for conducting business with customers and business partners over an "extranet. As indicated in the License Agreement when you download, you can freely use and redistribute the source code for these apps as desired. However, do not remove any proprietary copyrights or acknowledgements included by the developers. Please review the End User License Agreement for specific details. Who are these applications designed for? IS management can use the applications as prototypes or "proof of concept" applications that will give them some experience running SuiteSpot applications and help them to better understand how they work, what their performance characteristics are, and what specific functionality their organization may need. Departmental personnel or small companies can use the applications as prototypes, though they may also be able to actually deploy some of the applications. This could deliver important productivity improvements with no up-front cost. Departments and end users can also use the applications to show management what is possible or needed. This will speed up the decision-making process for deploying SuiteSpot-based applications. ISVs can use these applications as examples of the types of applications that can be built as well as to learn how Netscape SuiteSpot -based applications are developed and designed. This accelerates the learning process for ISVs, and in some cases the applets can serve as a starting point for viable commercial applications. This accelerates application development and stimulates ideas for other intranet applications. Where can I find a partner to help me develop my intranet? All of the developers providing applications through this program are available to help develop expanded versions of their products and for general development engagements. Please see the individual application descriptions for information about how to contact them directly, or visit the Web sites listed here:

Chapter 4 : Livewire "source" problem

Netscape LiveWire is a sophisticated visual development environment that enables professional Webmasters and Web developers to build and manage entire Web sites and applications.

Peter is a senior engineer with BTG Inc. He can be contacted at http: Developing sophisticated web applications that interact with databases is a complex task. In the past year alone, numerous toolsets have appeared that try to make this process easier. The tools typically take one of two approaches: They either require programming or generate code. In this article, I describe why I believe the programmatic approach to developing complex web sites is preferable to the code-generation approach. Traditional comparisons of software packages weigh issues such as execution speed, transaction time, executable size, and the like. The basis of my comparison is more pragmatic, however, focusing instead on the cost-effectiveness of web-site implementation and maintenance. Maintainable web systems are simple and modular, and their construction costs are minimized in both the short and long term. In the short term, these characteristics improve maintainability and save money by reducing complexity. Simplicity and modularity become even more critical when new programmers must understand and change the system without input from the original developers. Scalable systems reduce up-front hardware and database infrastructure costs and facilitate future growth. In a scalable system, porting from low-cost hardware to powerful, expensive hardware is virtually risk free because it involves no additional software development. In scalable systems, the same is true when porting from an inexpensive relational database to an expensive, industrial-strength database. Thus, scalable systems allow you to start with an inexpensive hardware and database solution and upgrade to a more powerful infrastructure when additional horsepower is necessary. To be truly scalable, a web site must also be extendable. That is, even though it may start small with simple functions, more complicated functions can be incorporated as the system grows. A truly scalable system is also, by definition, a high-performance system because it is able to scale up to powerful hardware and databases. The first two take the code-generation approach; the last two take the programming-language approach. Rather than write code, you specify appearance and functionality via GUI screens. The system then translates that specification to template files, query definition files, binding files, and HTML-generation code. The query definition files are in a proprietary format and capture graphically created database queries. The binding files, also in a proprietary format, map the template files to the HTML-generation code. During web-site run time, the code generates HTML that fills in the placeholders in the template files. In most cases, the user interface generates template files, query definition files, binding files, and HTML-generation code. Developers can also define these components directly in the underlying language: However, this forfeits the ability to automatically generate them in the future. Any visual web-page design tool can be used to generate the initial HTML, and any text editor can be used to embed the scripting-language statements. LiveWire and Cold Fusion require no special user interface for specifying web-site appearance or functionality. Both the HTML and scripting-language code are typically combined in the same file. The HTML defines the appearance. The scripting-language statements specify functionality and indicate, via SQL statements, how to retrieve and format database data. At run time, the system converts the scripting statements and retrieved data into HTML and sends it to the user. It also contains functions for accessing databases. Figure 1 provides an overview of the LiveWire architecture. Multiple database connections are established per application. Each connection can be shared among multiple users. A single ODBC connection is maintained per application. It is shared among multiple users. Application Servers generate one or more web pages. New Application Server processes are spawned for each concurrent user accessing the system. They shut down after a configured amount of time or number of web transactions. The Application Server Manager is a background process that schedules and manages Application Servers. Finally, Figure 4 presents the WebObjects architecture. As it is a CGI and not memory resident, it lives for the duration of one web transaction. Application executables generate one or more web pages. New Application Server processes are created by the Adapter Server for each concurrent user accessing the system. Database connections are maintained for the life of an application executable. Toolset Comparison Figure 5 depicts how the cost effectiveness of a system design is based on maintainability and scalability and, likewise, how maintainability and scalability are based on design characteristics such as simplicity, modularity, portability, extendability, and performance. As Figure 5 illustrates, these are also desirable characteristics in web-site design. Simplicity makes life easier for the developer -- the person who must maintain or extend a system built by someone else. To be simple, a web-database site should have few components, few concepts, and code that is easy to follow. LiveWire and Cold Fusion contain few components. LiveWire and Cold Fusion not only have minimal internal components, they also do not rely on external compilers, special libraries, and other products. New versions of these external products could bring about new bugs or incompatibilities, and may cause portability problems. WebObjects has a built-in compiler that reduces this problem, but adds a different kind of complexity by supporting two styles of its proprietary WebScript language and, in a future release, JavaScript. LiveWire and Cold Fusion involve few concepts, and the basic concepts necessary to develop a system are all specified in the same file. Embedded scripting-language code is easier to follow than automatically generated code. Developers unfamiliar with a particular web system built with LiveWire or Cold Fusion need only look in one file to determine how a particular web page operates. Formatting, control flow, and data access are all specified in the same place. In contrast, LiveWire and Cold Fusion involve human-generated code. The driving force behind using an application generator is to simplify development by reducing the need to manually write code. Also, coding will still be necessary for the parts of the system that cannot be generated automatically. Ultimately, you will have to become expert at both a complex, proprietary user interface and at reading and modifying machine-generated code. This is not the case for programming-language-based web development, since there is no special user interface to learn and the code is high level and human generated. Being able to compartmentalize the underlying code and data for a web system makes the code easier to understand and facilitates reuse. The most modular web-database systems minimize code and data replication and separate look-and-feel from functionality. LiveWire minimizes code replication by allowing you to define JavaScript functions. Functions in JavaScript, as in most modern programming languages, can be simple or complex and can be used throughout the system. Cold Fusion does not contain functions or similar constructs to prevent code replication. Data replication occurs in web-database systems that copy database data to temporary HTML files. This could become a maintenance nightmare, because data duplication occurs in many places and data could become unsynchronized. When look-and-feel is distinct from functionality, a graphic designer can modify page layout and graphics without having to use a compiler or code in C. This is a powerful characteristic because it allows splitting web-site development work naturally between two very different specialists. LiveWire and Cold Fusion support this capability because a graphic designer can modify and extend HTML components using the visual editing tool of his or her choice without touching the uninterpreted scripting code. Consequently, the graphic artist would ultimately need to learn how to use the same tools as the code developer and would not be able to use his or her favorite visual HTML editor. This contributes to a more readable and, therefore, maintainable system by more closely equating the code structure with the structure and control flow seen by the web user. Sophisticated web-database systems should also support ODBC and should be portable between the major industry-standard database systems. The ability to scale up functionality is just as important as the ability to scale up hardware, operating systems and databases. In other words, an existing site, however simple, should always be extendable to new, sophisticated features. Web-database software can go a long way toward extendability if it can display database data in any format allowed by HTML. Rather, they contain only placeholders that are filled in at run time by a C or WebScript function. If an automatically generated function does not exist to display data in a given manner, then it must be written by hand. This is not simple because it requires understanding how to interface with the existing machine-generated code. Automatic code generators cannot possibly be designed to predict and handle all types of functionality that might be required. Only human code generators can do that. If a web-database system is portable, then it is usually possible to improve performance by "throwing hardware" at the system. Nevertheless, it is cheaper and obviously preferable to improve performance by using more efficient software. Two performance-enhancing tricks used by web-database software are to integrate with the web-server process and to maintain database connections. Integrating web-database run-time managers with web servers reduces overhead. So, although there is an additional process, it runs continually and is shared for multiple web accesses. This can become expensive when many users access the site at the same time. Maintaining database connections is more efficient than reestablishing them with each access. LiveWire illustrates this performance difference because it supports both maintained and reestablished connections. In the maintained-connection mode, a definable number of database connections are opened when the web server starts. Data access speed improves substantially because there is no connect and disconnect overhead. The downside to the maintained-connection mode is that the same database account must be shared by all users. Systems that require different database accounts for different users must use the serial approach of reestablishing the connection for each access.

Chapter 5 : Ted Coombs - Wikipedia

Find helpful customer reviews and review ratings for The Netscape LiveWire Sourcebook: Create and Manage a Java-Based Web Site at calendrierdelascience.com Read honest and unbiased product reviews from our users.

Chapter 6 : Netscape AppFoundry FAQ

The Netscape LiveWire Sourcebook: Create and Manage a Java-Based Web Site by Ted Coombs,Jason Coombs,Don Brewer. John Wiley & Sons, Paperback,ex-library, with usual stamps and markings, in good all round condition.

Chapter 7 : LiveWire Web-to-Database Solutions, Part 1 | Dr Dobb's

INFORMIX-OnLine Workgroup Server is bundled with Netscape LiveWire Pro, Netscape SuiteSpot and is available in Netscape Payment Kit and Netscape AppFoundry applications. Informix Makes "Promise of the Web" a Reality with New Universal Web Architecture; INFORMIX-Universal Web Architecture Provides a Flexible Web Environment for Fast, Easy.

Chapter 8 : Oracle iPlanet Web Server - Wikipedia

The Netscape Livewire Sourcebook: Create and Manage a Java-Based Web Site. Ted Coombs. from: N/A.

Chapter 9 : Server-Side JavaScript Guide

Initially a separate product, LiveWire was included starting with Version of Netscape Enterprise Server. LiveWire Pro includes an Informix database and the Crystal Reports report and analysis.