Login

PRODUCTS ◦ SALES ◦ SUPPORT ◦ DOWNLOADS ◦ ABOUT

Home » Products » Elevate Web Builder - Rapid Application Development

Elevate Web Builder - Rapid Application Development

Do you need your application to be web accessible, but don't want to have to deal with hand-coding the scripts and markup ?

Elevate Web Builder is a 100% visual rapid application development tool for client-side web applications. It is simple and easy to create full interactive web applications by simply dragging and dropping components on to forms, and then changing their appearance and behaviors as desired.

There is absolutely no need to learn JavaScript in order to use Elevate Web Builder. The product uses a compiler to compile standard Object Pascal source code into JavaScript, emitting all necessary scripts, loader markup, images, and style sheets, and then quickly and easily deploying your application to the destination of your choosing. This means that you can continue to leverage existing language skills and take advantage of the many optimizations that the compiler can provide compared to hand-coded JavaScript solutions. The compiler provides many features that are present in Object Pascal, but not explicitly supported in JavaScript, such as formal inheritance with virtual methods/overrides, constructors and destructors, object scopes (private, protected, public), unit interface and implementation scope, and proper method pointer scope for event handlers, to name just a few.

Do you dread having to target multiple browsers and ensure that they all look and function the same ?

Web applications built with Elevate Web Builder always look and behave exactly the same way, regardless of whether they are running in Internet Explorer, FireFox, Opera, Chrome, or Safari. Elevate Web Builder includes full theming support and all visual controls are themed and appear exactly the same, down to the pixel, on any modern web browser. Themes can be switched at runtime, allowing you to tailor the interface of the application to the type of client operating system in use, whether it be a cell phone or a desktop PC.

Do you want to develop web applications in the same manner as desktop applications ?

Elevate Web Builder brings the desktop integrated development environment that we are all familiar with to web applications. All requests to the web server are made dynamically, and you can completely control how the entire application responds to keystrokes, mouse clicks, errors, and much more. Complete source code is provided for the entire framework used in the product, so you are always free to customize or improve upon existing controls and functionality. In addition, you always have complete access to all of the underlying web browser functionality that the framework relies upon.

Find out more information on Elevate Web Builder:

Elevate Web Builder for Web Application Developers »

This web page was last updated on
Monday, January 21, 2013 at 02:36 PM

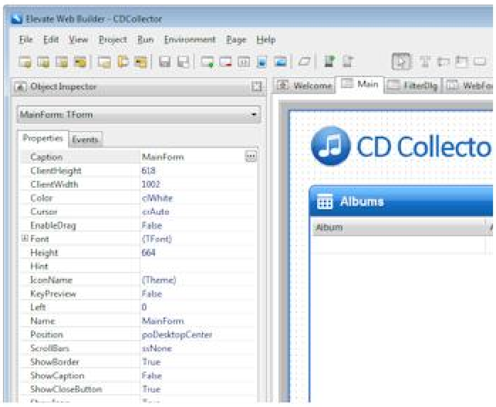
PRIVACY POLICY ◦ SITE MAP

Valid HTML 4.01 Valid CSS

Questions or comments ? E-mail us at info@elevatesoft.com

Elevate Web Builder for Web Application Developers

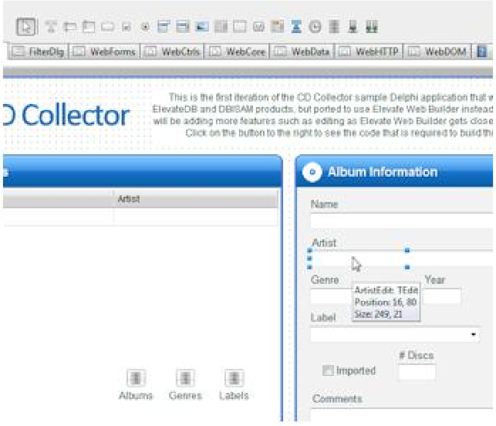
Web Development Productivity



Caption	MainForm
ClientHeight	618
ClientWidth	1002
Color	clWhite
Cursor	crAuto
EnableDrag	False
⊞ Font	(TFont)
Height	664
Hint	
IconName	(Theme)
KeyPreview	False
Left	0
Name	MainForm
Position	poDesktopCenter
ScrollBars	ssNone
ShowBorder	True
ShowCaption	False
ShowCloseButton	True

Traditionally, development of web browser applications has been a very tedious process that involves a lot of hand-coding, and experience in several different, but necessary, technologies. Desktop application developers have had integrated development environments (IDEs) for years that combine both the design and coding aspects of applications into one unified tool that allows a developer to be very productive. Elevate Web Builder brings this productivity to web browser application development, allowing developers with a lot of desktop application development experience but less web development experience, to instantly create web browser applications. More importantly, it enables the developer to create web browser applications that are guaranteed to contain fewer bugs, and therefore be much more reliable than any hand-coded alternative. A web browser application created with Elevate Web Builder will look the same in all modern browsers, relieving the application developer of the very difficult job of ensuring that all visual controls in their application look and behave the same way. All visual controls are themed, and applications can use any number of themes and can switch themes at runtime.

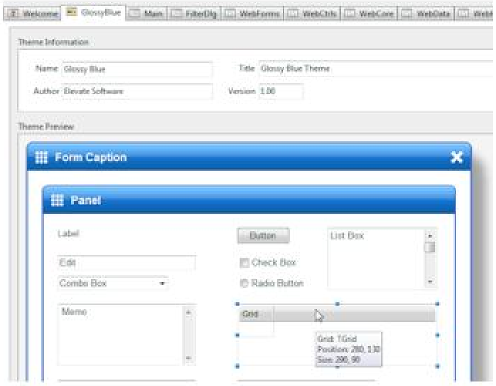
Elevate Web Builder's integrated development environment contains the following key features:



This is the first iteration of the CD Collector sample Delphi application that w
ElevateDB and DBISAM products, but ported to use Elevate Web Builder instead
will be adding more features such as editing as Elevate Web Builder gets close
Click on the button to the right to see the code that is required to build th

ArtistEdit: TEdit
Position: 16, 80
Size: 249, 21

- A **very** fast compiler that compiles the source code of any project into an optimized JavaScript application.
- An integrated, WYSIWYG (What You See Is What You Get) form designer and object inspector that allows the developer to design the visual elements of a web browser application quickly and easily.
- An easy-to-use code editor with syntax highlighting and automated code updating to reflect changes in the form designer and object inspector.
- A project manager that provides quick access to all source units and external source code in a project.



Name: Glossy Blue
Title: Glossy Blue Theme
Author: Elevate Software
Version: 1.00

Grid: TGrid
Position: 280, 130
Size: 290, 90

- A local web server and browser for running your web applications directly in the IDE. The local web server also allows for sending debug messages directly to the IDE.
- A theme editor for creating new themes that can be distributed to other Elevate Web Builder developers.
- Easily-accessible online help that loads directly in the IDE and does not have any external dependencies that can easily break and make documentation unavailable.

Extensive Component Framework

Elevate Web Builder includes a complete component framework that includes both visual components and non-visual components. The following components are available for use on a form:

Order

Ordering is fast, easy, and secure, and you can be installing Elevate Web Builder within the hour. Every Elevate Web Builder purchase includes a 30-day money-back guarantee.

Order Now »

Trial Version

You can download a trial version of Elevate Web Builder and evaluate the product before your purchase.

Download Now »

Operating Systems

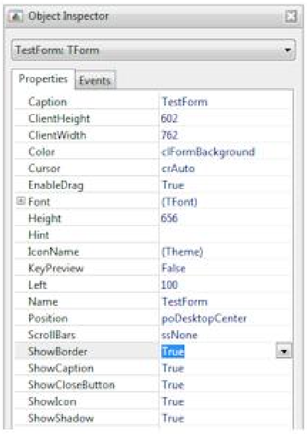
- Windows XP or higher

Component	Description
TLabel	Static text control
TEdit	Single-line edit control
TMemo	Multi-line edit control
TButton	Captioned button control
TCheckBox	Check box control
TRadioButton	Radio button control
TComboBox	Combo box control
TListBox	List box control
TImage	Image dipslay control
TGrid	Grid control
TPanel	Panel control
TSeparator	Separator control
TPlugin	HTML plugin control
TPage	HTML page control
TLink	HTML link control
TTimer	Timer component
TDataSet	Dataset component
TServerRequest	Server request component
TServerRequestQueue	Server request queue component

The framework includes many additional functions, procedures, and classes to use for object lists, string lists, parsing and string manipulation, as well as direct manipulation of the browser DOM (Document Object Model) classes, functions, procedures, and variables.

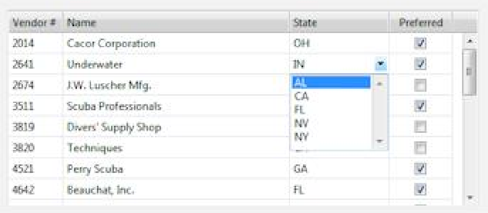
The Elevate Web Builder framework also includes the following notable features:

- Forms and panels can look like windows, or they can look like normal web "pages". By default, forms and panels have caption bars, icons, close buttons, borders, and shadows, but any of these features can be turned off with a simple property switch at design-time. Dragging of forms and panels is also supported through a single property but, by default, is not enabled.
- Forms can be shown normally in an asynchronous manner, or they can be shown modally. If a form is shown modally, then the user will not be able to interact with any other visual portion of the application until the form is closed. The framework also includes standard modal dialog methods that can be used to display error messages or confirm certain operations with the user.
- All visual components are themed, and their look and feel can be completely changed by simply changing the active theme for the application at either design-time or run-time.



Caption	TestForm
ClientHeight	602
ClientWidth	762
Color	clFormBackground
Cursor	crAuto
EnableDrag	True
Font	(TFont)
Height	656
Hint	
IconName	(Theme)
KeyPreview	False
Left	100
Name	TestForm
Position	poDesktopCenter
ScrollBars	ssNone
ShowBorder	True
ShowCaption	True
ShowCloseButton	True
ShowIcon	True
ShowShadow	True

- All focusable visual components have different visual states for when the mouse is over the control, when the control has focus, or when the control is clicked.
- All visual components share a common set of mouse events for mouse down/up, move, enter/leave, click, and double-click event handling. All edit components include additional keyboard events for key down/up, and key press event handler. All focusable components include additional enter/exit events.
- Most visual components can be unbound (the default state) or bound to a dataset and column. This avoids a situation where there are separate versions of the same control for unbound or bound usage. Also, this gives the developer the ability to mix unbound and bound controls as necessary. When a visual component is bound to a dataset and column, its contents are automatically updated as the dataset is navigated and/or modified.
- The grid component is fairly extensive, and supports mixing unbound and bound columns, individual cell selection or entire row selection, the display of column headers, in-place editing via edit, combo box, or check box controls, read-only and disabled columns, column header clicking (including automatic column sorting for grids bound to datasets), and fixed or sizable columns.



Vendor #	Name	State	Preferred
2014	Cacor Corporation	OH	☑
2641	Underwater	IN	☑
2674	J.W. Luscher Mfg.		☐
3511	Scuba Professionals		☑
3819	Divers' Supply Shop		☐
3820	Techniques		☐
4521	Perry Scuba	GA	☑
4642	Beauchat, Inc.	FL	☑

- Datasets are easily handled with the dataset component, which supports navigation, searching sorting, inserting, updating, deleting, transactions, and two-phase commits. Dataset columns can be defined at design-time or loaded from a JSON (JavaScript Object Notation) string that was returned from a web server using a server request. Likewise, dataset rows can be loaded from a JSON string.

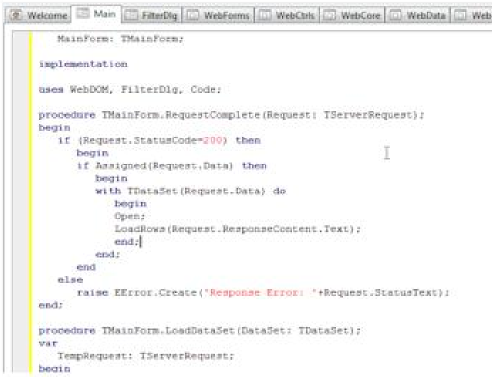
- A server request component is used to communicate with the web server in an application. All server requests are asynchronous, but a server request queue component is also provided in order to ensure a serialization of requests for a particular resource. Server requests can send/receive any textual data, including XML/HTML, JSON, key-value pairs, or base64-encoded binary data.

Language and Compiler Features

The language used by Elevate Web Builder is an Object Pascal dialect that retains as much compatibility as possible with the Object Pascal language used by the Delphi development environment from Embarcadero Technologies. Object Pascal is a highly-structured, strongly-typed, object-oriented language that is also simple to use due to its English-like keywords and statements. This makes it an ideal language for projects involving multiple developers where all developers working on the project must be able to easily read the source code of other developers. The compiler in the Elevate Web Builder IDE takes advantage of the strongly-typed nature of the Object Pascal language to only emit JavaScript code in the emitted application that is actually referenced. Any functions, procedures, constants, variables, types, classes, member variables, methods, or properties that aren't referenced in the source code for an application are not emitted. This keeps the size of the resultant application as small as possible.

Some of the more interesting features of the language and compiler are:

- The compiler can emit a compressed (and obfuscated) application that makes the application very small and very difficult to read with the naked eye or reverse-engineer into any meaningful source code. Compiler compression can be toggled on or off in the project options for any project, and a compressed application is normally less than half the size of an uncompressed application.

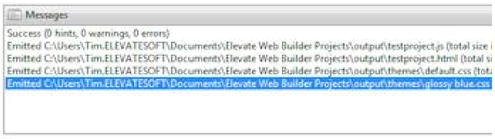


```
   MainForm: TMainForm;

implementation

uses WebDOM, FilterDlg, Code;

procedure TMainForm.RequestComplete(Request: TServerRequest);
begin
   if (Request.StatusCode=200) then
      begin
      if Assigned(Request.Data) then
         begin
         with TDataSet(Request.Data) do
            begin
            Open;
            LoadRows(Request.ResponseContent.Text);
            end;
         end;
      end
   else
      raise EError.Create('Response Error: '+Request.StatusText);
end;

procedure TMainForm.LoadDataSet(DataSet: TDataSet);
var
   TempRequest: TServerRequest;
begin
```

- The compiler includes a very small runtime overhead of around 10KB for an empty, and uncompressed, non-visual application, and around 4KB for an empty, but compressed, non-visual application.

- The compiler emits all files necessary to successfully run the application, including an HTML loader file, the JavaScript application, all included theme style sheets, and any resources such as images that are specified at design-time.

- The compiler can display automatic warnings and hints for source code issues such as uninitialized variable references or variables that are declared but never referenced.



```
Success (0 hints, 0 warnings, 0 errors)
Emitted C:\Users\Tim.ELEVATESOFT\Documents\Elevate Web Builder Projects\output\testproject.js (total size
Emitted C:\Users\Tim.ELEVATESOFT\Documents\Elevate Web Builder Projects\output\testproject.html (total si
Emitted C:\Users\Tim.ELEVATESOFT\Documents\Elevate Web Builder Projects\output\themes\default.css (tota
Emitted C:\Users\Tim.ELEVATESOFT\Documents\Elevate Web Builder Projects\output\themes\glossy blue.css
```

- The String, Integer, Boolean, DateTime, class, and method types are supported. Strings are immutable and cannot be modified in-place. The DateTime type is type-compatible with the Integer type and represents raw milliseconds, making date/time arithmetic fairly easy. Method type references, such as those used for events, are referenced and de-referenced automatically by the compiler based upon how they are used, eliminating the need for special address-of or de-referencing operators.

- A base TObject class exists in the runtime that is the ancestor of all classes. All classes use a normal Create/Free lifecycle, and the Free method automatically checks for a nil reference when executing, so a procedure such as FreeAndNil is not required.

- Class methods can be declared as **virtual** and overriden by descendant classes with the **override** keyword. Methods can also be declared as **abstract**, and will trigger a compiler error if an instance of a descendant class is created without the abstract methods being overriden and implemented.

- Static class variables, methods, and properties are supported.

- All functions, procedures, and methods can have default parameters, and all variables can have default values. Variables that are declared within a class can also have default values, and any instance of the class will automatically have all of its variables initialized to the specified default values. If a variable within a class doesn't have a default value, then the compiler will automatically generate an appropriate default value based upon the type of the variable.

- Dynamic arrays can be passed directly as constant parameters to any function or procedure without being created first. Dynamic arrays can also be declared with default values, thus eliminating the need to specifically initialize them with values.

- Overloaded functions, procedures, and methods do not require a separate **overload** keyword.

The compiler automatically figures out which version of any particular function, procedure, or method to call based upon its signature (parameters, result type).

- Indexed properties can be designated as default properties with the **default** keyword, and there can be multiple default indexed properties that accept different index types. The compiler automatically figures out which version of the property to use by its index parameter type. In addition, arrays can be used directly as indexed properties without requiring getter/setter methods.

Roadmap

The following is a list of features and improvements that we are currently working on for Elevate Web Builder, their scheduled implementation version (if available), and their completion status.

	Scheduled	✓ Completed
Compiler		
Command-line compiler	1.03	
Improved compilation of internal system functions (prevent scope inversion)	1.03	
Framework		
HTML5 canvas support (TPaintControl)	1.02	
Tabbed notebook control	1.02	
Menu control	1.02	
Anchoring and automatic flow/layout of controls	1.03	
Date/time picker control	1.03	
Web Server		
External, deployable HTTP server that supports dataset JSON handling and static content	1.01	✓ 1.01
Pluggable, native-code modules for external HTTP server	1.01	✓ 1.01

More Support Options

- Support Plans

- Support Forums
- Frequently Asked Questions
- Technical Bulletins
- Technical Articles

- Incident Reports

- Product Manuals

This web page was last updated on
Wednesday, January 9, 2013 at 06:53 AM

Valid HTML 4.01 Valid CSS

Questions or comments ? E-mail us at info@elevatesoft.com