

# eXtc Web Developer

## Overview

Version 3.0.599

---

### What is eXtc Web Developer?

eXtc Web Developer (EWD) is a very high productivity, enterprise-grade platform for developing web applications. It includes an advanced but extremely easy to use Ajax Framework, allowing sophisticated, modern web applications to be built almost as quickly and simply as static web pages.

EWD has been designed to be technology-independent, and is potentially capable of working with any web application front-end environment (eg PHP, Java Server Pages, ASP.Net), any scripting language and any database.

### Background

There are numerous technologies for developing web applications, all of which have their pros and cons, and each with their own supporters and detractors. There are mainstream, non-proprietary and/or Open Source technologies such as PHP and Java Server Pages (JSP), and there are proprietary technologies such as .Net. The choice of technology will usually be a result of issues such as performance, cost, available skills, corporate strategy or independent technical advice. Although the end product of these technologies is HTML and JavaScript, the mechanics and methodology needed to program and use these technologies are very different. It is very difficult to change your mind and change technologies once you have committed to one approach. Programming staff will tend to be skilled in the technology you've chosen, and changing technologies involves significant retraining and/or re-recruitment of staff.

eXtc Web Developer (EWD) completely changes this situation and allows you to develop your web applications in a way that is independent of your web application run-time technology, and allows you to focus on what you are trying to do, rather than how you'll do it. EWD's primary design goal is to radically improve the efficiency and productivity of your web application development team. EWD allows you to describe your web applications in a highly maintainable and easily understandable format. EWD will generate all the run-time code for you for the front-end technology you select.

### Technology Neutral

EWD is unique in the Web Application framework marketplace because it is totally technology-neutral. Its architecture and technology has been designed to be completely flexible, and is potentially capable of generating web applications that will run on any technology at any of the web application tiers. The current version of EWD (3.0.599) can work with any combination of the following technologies:

- “front-end” technology
  - PHP
  - Java Server Pages
  - Caché Server Pages
  - WebLink
- Session persistence database
  - Caché
  - GT.M
- Application scripting:
  - Caché ObjectScript
  - MUMPS
- Application database
  - Caché
  - GT.M

However, we already have a versions under development that are extending this to include:

- “front-end” technology
  - ASP.Net
  - Ruby
  - Python
  - Perl
- Session persistence database
  - mySQL
  - SQL Server
  - Oracle
- Application scripting:
  - PHP
  - Java
  - C#
  - Ruby
  - Perl
  - Python
- Application database
  - mySQL
  - SQL Server
  - Oracle

As you can see, EWD is potentially a framework for use with any of the standard web application technologies. Its ability to cater for all these platforms stems from its unique XML-based compiler, and its carefully designed architecture.

This overview document will focus on the currently supported technologies. If you are interested in helping us to develop EWD for other platforms, contact Rob Tweed ([rtweed@mgateway.com](mailto:rtweed@mgateway.com))

## **eXtc Web Developer: A summary**

EWD is a framework for web application development. It also includes a framework for Ajax application development.

EWD has been designed to be as simple as possible to use, and yet allow you to build web applications that are as complex and sophisticated as you require. EWD focusses primarily on the design aspect of web applications, and, by automating all the key processes of a web application, reduces the programming requirement of a typical web application to almost trivial levels. Once you become familiar with EWD, you'll wonder why every other web application development tool in the marketplace makes it all so complex!

Not only is EWD designed to allow you to create web applications more quickly than any other framework, but also it is designed to massively reduce the maintenance overhead of those applications – something that is almost always overlooked in other frameworks. EWD has been carefully designed to maximise the productivity of your development team.

A major part of this productivity gain results from EWD allowing your developers to focus on *what* your applications are supposed to be doing, rather than *how* they'll do it.

### **“What” versus “How”**

Anyone who has used standard development technologies to build web applications (eg PHP, Java Server Pages, ASP.Net, Caché Server Pages etc) will know that the task is complex, and not something that could be attempted by anyone other than a programmer. There are a number of higher-level tools available (eg content-management-based systems such as PHP Nuke or Ektron, and template-based systems such as Smarty). However, they are nevertheless programmer-oriented tools, introduce new additional syntax and concepts, in addition to the already complex combination of HTML, JavaScript, CSS and Java, PHP or .Net scripting, and/or they force you to work in a particular way and with particular technologies. PHP Nuke, for example, is pretty much tied to the MySQL database, and Ruby on Rails requires you to adopt the Ruby language.

With any high-level tool, there is the additional risk that you end up expending most of your time and energy fighting to make it work the way you want it to, rather than being forced to work in the way in which it wants you to work.

Whatever technology you use, you'll find two common threads:

- The task of building a web application is a complex one, requiring the skills of a programmer. Far from making life simple, each new generation of development tools seems to add yet more complexity, add new concepts and perpetuate the view that web application development is a complex, technical, programming exercise;
- The majority of the programming effort involved relates to the *how* – including all the mechanics of integrating the web browser front end with the back-end database. The result is a maintenance headache – give ten programmers the task of building even a simple database-linked web form and you'll end up with at least ten different approaches. Then try getting another programmer to understand *what* someone else's page is trying to do. The programmer will spend a lot of time wading through acres of embedded programming logic, unable to “see the wood for the trees”. It is this inextricable spaghetti of *what* and *how* that makes web application development complex, time consuming and leads to maintenance headaches. The more web applications you develop and the larger your applications get, the worse this maintenance headache becomes until all your available resources are tied up maintaining existing pages, leaving no room for new development!

## The eXtc Web Developer Philosophy

The philosophy behind eXtc Web Developer is to break the mould and escape from the problems inherent in the mainstream approach to web application development. In our experience, there is no excuse for web application development not to be simple and the programming involved in even the most complex web application should be suprisingly trivial. Indeed, web application development should be predominantly a design-oriented task, not a technical programming-oriented exercise at all. This is perhaps not something a professional programmer wants to hear, and flies in the face of the ever more complex web application development environments produced by the IT industry, but the fact is that EWD allows programmers to focus on what's really important in those web applications instead of wasting time writing logic that is inherently capable of being automated for them.

Our approach and philosophy is well founded and is based on over 10 years of research, practical web application development and support of major web applications around the world. Web applications developed and used by our customers include some of the largest, most demanding and sophisticated in the world.

It is eXtc Web Developer's ability to separate the *what* from the *how* in web applications that makes it so powerful. It allows you to build your application as if it was a storyboard of simple HTML pages, with a simple HTML-oriented way of describing what you are wanting to do. eXtc Web Developer looks after the *how* for you, by transforming your Design Pages into the actual run-time version for you. All the necessary security and session management logic is created automatically, as are all the linkages that allow the "front-end" technology (eg PHP, JSP etc) to correctly interact with the back-end database (eg Caché or GT.M) .

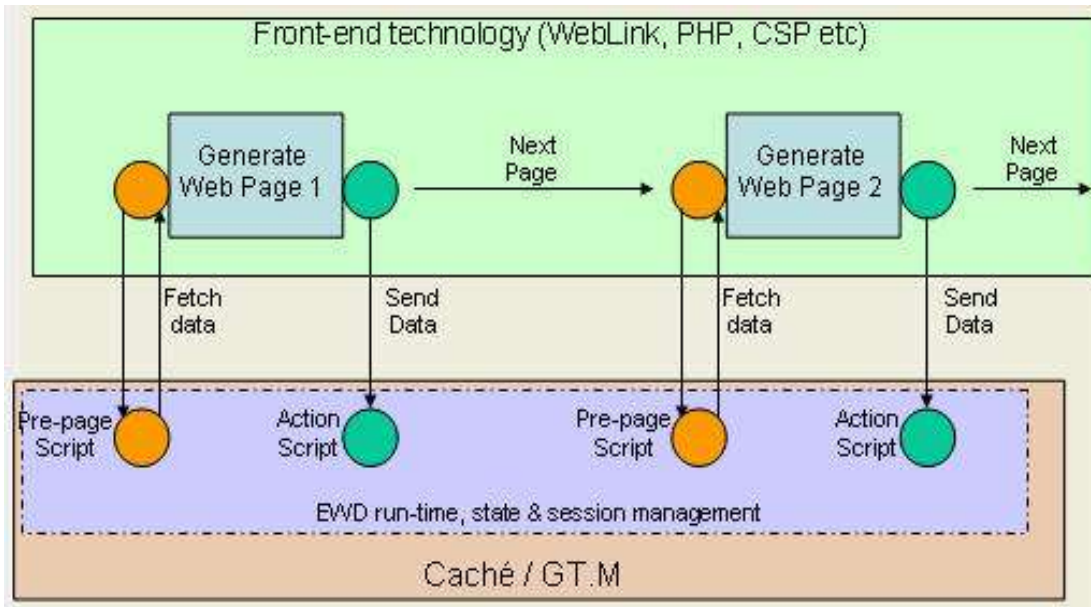
Sounds too good to be true? Let's examine how eXtc Web Developer works in some more detail and you'll discover just how powerful and effective it really is, and how it is revolutionising web application development for our customers.

### **The Storyboard Approach**

The user experience – the design, look and feel of the web pages that they'll see in your application – is without doubt *the* most critical aspect of any web application.

If it wasn't for the need to link a web application to data held in a database, a web designer could build an entire application as a series of static HTML pages. This is the conceptual starting-point for an EWD application : think of your application as if it was a set of HTML pages – a *storyboard*. Focus on the user interface and get your web designer to construct the storyboard for the application, in other words the set of individual web pages that a user will experience when they use your application. Some of the transitions between pages will be hard-wired, eg through hyperlinks, but others will be determined at run-time in the final application, dependent on the data and/or the responses entered by the user. Don't worry about this at this stage – as you'll discover, EWD makes this easy.

Turning this storyboard (known as EWD Design Pages) into a working application actually only needs the addition of a small number of things, but to clarify what these are, let's examine the interactions that occur between the browser and back-end database in *any* web application. Perhaps surprisingly, *every web application adheres to the exact same sequence of events*, summarised by the figure below.



The steps are as follows:

- Prior to sending the web page to the browser, any data needed for that page is retrieved from the database and transformed, if necessary into the correct structure for display in the browser. In EWD, the method that is used for this task is known as the *pre-page script*.
- During the rendering of the web page – ie its programmatic generation within the *front-end* technology (PHP, JSP etc) – there may be conditional or looping logic that needs to be added to generate the correct HTML and other markup. For example, the rows in a table may need to be generated using data retrieved in the *pre-page script*. In EWD, such pieces of logic are known as *in-page scripts* and are defined in a technology-independent way using XML-based custom tags embedded in your Design Pages.
- Once the user has the web page in their browser, there are only four things that can possibly happen:
  - Nothing at all – the user ignores the page and moves on to something else;
  - Log out – the user chooses to leave the application;
  - A hyperlink is clicked (or more generally an HTTP GET request is sent) ;
  - A submit button is clicked (or more generally an HTTP POST request is sent).

In the latter two cases, the only real practical difference is that when a submit button is pressed, the user will have usually entered or edited some data in a form, and the contents of that form will be sent as a series of user-defined name/value pairs ; when a hyperlink is clicked, the data being sent (ie the name/value pairs) will be predetermined.

- In the case of the form being submitted, you'll want to do the following things:
  - Validate the name/value pairs that were submitted;
  - If they pass validation, process the name/value pairs in some way, eg transform and save them into the database
  - If they fail validation, then you'll want to create an error message that will be displayed in some way to the user.

In EWD these processes are handled by *action scripts*.

- You'll then want to take the user to the next page:
  - if they clicked a hyperlink, the next page is predetermined.
  - if they clicked a submit button, but the data failed to validate, you'll probably want to put the data entry page back again and give them some visual warning of the error that was identified;
  - however, if the data passed its validation, you'll want to take them to the next page in the storyboard, either to a predetermined page, or to one that is determined by the data they entered.
- The cycle then repeats.

It's obvious, when you think about it, that all web applications work in this way. But you'll find that every other framework masks this simple cycle. By comparison, EWD's run-time architecture is firmly based around this model, and if you visualize your web application in this way, you'll find that EWD massively simplifies your development.

### **Other productivity features**

Let's quickly summarise some of EWD's other productivity features. You'll find full details about how to use them in the other documentation and training videos.

- Fully automated security and session management

Unlike other web development technologies, EWD automates all aspects of session, state and security management. You just define the links between your EWD Design Pages as simple references, in effect as if they were links to static web pages. These are automatically expanded by EWD's compiler to include all the necessary tokens that are used for EWD's high security run-time regime. The result is EWD Design Pages that are not cluttered with huge amounts of state and security management code, making them much more understandable and maintainable.

EWD's automates the maintenance of your application's persistent session data. By automatically persisting name/value pairs and form fields wherever it is safe to

do so, EWD once again reduces the code that would otherwise clutter and confuse your Design pages.

- Form Field session mapping

EWD automatically maps HTML form fields to special Session *data containers*. Checkboxes and Radio Buttons can be automatically pre-checked, <select> tags drop-down options pre-populated and highlighted, and text fields and textarea fields pre-populated by simply setting the mapped Session *data containers* in your pre-page scripts. Forms in your Design Pages no longer need to contain any logic to define how they are pre-populated or pre-checked, making them easier to understand and maintain.

When you submit HTML forms you'll find that the same logic is applied automatically in reverse. The submitted name/value pairs are mapped automatically back to the corresponding Session *data containers*. It all adds up to less code in your Design Pages and back-end scripts, resulting in applications that are easier to understand and easier to maintain.

- Templates

A particularly powerful feature of eXtc Web Developer is its templating capabilities. *Template* documents and *include* files allow you to separate out and modularise the common look and feel aspects of your web application's pages.

- Custom tags

You can further modularise your applications and extend EWD's functionality by designing your own XML-based custom tags. These allow you to build complex re-usable widgets for your applications that can be dropped into any of your pages. Custom tags allow you to avoid the maintenance problems produced by programmers cutting and pasting "cool" code between pages and then making different modifications to each instance. Once you learn how to create EWD Custom tags, you'll find that there is literally no limit to the complexity and power of the tags you can create.

- Ajax framework

EWD allows you to use the most modern Ajax techniques for your web applications. EWD's amazingly simple yet totally flexible Ajax framework will change the way you think about the design and look and feel of your web applications. No other framework makes Ajax so conceptually simple. EWD brings its very rapid development and ultra low maintenance into the hands of the Ajax developers!

- Multi-lingual Localisation
  - EWD provides a very simple yet effective solution to a perennial problem – how to localise web applications without adding yet more complexity to your development cycle.

With EWD, your web application designers and programmers continue to develop and modify the Design Pages in their native language. EWD's compiler can optionally detect all the text it finds in the Design Page, automatically add it to a look-up dictionary database and replace the text in the page with a procedure call that will retrieve the text from the look-up dictionary in the appropriate language if it's available (defaulting to the native language if not). EWD provides utilities for exporting the dictionary database for translation, and for importing the translated translations. It even provides a run-time mode that aids the translator who is performing the contextual review of the application's translated pages.

### **Developing Web Applications using eXtc Web Developer**

The process of developing a web application using EWD breaks down to the following steps:

- Create your high-level EWD Design Pages
- Create any associated pre-page script and action scripts for each page and add the references to these scripts into your EWD Design Pages
- Decide the technology in which you want your application to run (PHP, Java Server Pages, CSP or WebLink)
- Compile the application using EWD's XML-based compiler
- Your application is ready to run. Just point a browser at the first page in the application and away you go! Consult the other documentation or the video training guides to find out how to determine the correct URL to use
- To make modifications to your application, edit the appropriate EWD Design pages, recompile and run again.

### **Next Steps**

If you want to get started with EWD, you'll need to download it from our web site and then follow the installation and configuration instructions on our web site (<http://www.mgateway.com>). You'll also find examples and documentation there.

If you want to create PHP or Java Server Pages applications using EWD, you'll also need to download and install our M/Gateway Service Integrator (MGWSI). Once again you'll find this at our web site.

EWD and MGWSI are provided free of charge, but without any support. If you're interested in purchasing support for either product, contact Rob Tweed ([rtweed@mgateway.com](mailto:rtweed@mgateway.com)).

We hope that you'll find EWD enjoyable to use, and we are confident that you'll quickly appreciate its ability to simplify the development and maintenance of your web applications. Quite simply, there's nothing else like it!