1

Introduction

1.1 Is this book for me?

This book is not for beginners. It does not cover XHTML, HTTP, and all those other standards on the web; it requires the knowledge of them. This book is for web developers. No matter whether you are migrating from Zope 2 or have prior experience in competing technologies, like J2EE or Vignette StoryServer, or even Python-based frameworks like Webware and CherryPy – this book is for you! It certainly is not if you are looking for a thorough introduction to web applications developing.

This book is not a cook book. It does not present recipes for specific tasks nor solutions for specific problems. Rather, it introduces the web applications developer step-by-step to the world of Zope 3 and its component architecture and tries to give the reader the necessary skill for building web applications. It does not cover advanced tips and tricks such as fine-tuning applications or making applications scale better in large environments.

1.2 What is Zope?

Zope is

- a collection of free software
- jointly developed by Zope Corporation and a large community of software developers
- that you can use in whole or in part
- to manage complexity in gluing software components together,
- securely publish objects on the web and other systems,
- and make it easy to do Quality Assurance.

Steve Alexander at EuroPython 2004

a collection of free software: Zope is freely distributable software, it comes free of charge. That does not mean it does not have a value, it just means that you do not have to pay to use it. This successful concept is commonly called *Open Source*. Please consult the website of the Open Source Initiative for a definition of Open Source [7] and the Zope community website for a copy of the *Zope Public License* [21] under which Zope may be distributed.

jointly developed by Zope Corporation and a large community of software developers: After Zope Corporation opened Zope 2's source code to the public, it also started propagating an open development model, encouraging other developers to contribute. Zope 3 too has been a community effort from the beginning, with the majority of the contributions coming from a large community of software developers around the globe. The benefits for the single developer, the whole community as well as the quality of the software have proven to be enormous.

that you can use in whole or in part: Zope is very modular. It is possible to use just certain functionality from Zope and it is not even necessary to install all parts of Zope. Zope is a *collection* of many software modules.

to manage complexity in gluing software components together: Requirements for web-based applications are increasingly complex and the need for systems that can handle such complexity is growing. Zope solves this problem elegantly by separating responsibility into many different components. The way these components are then "glued together" determines the behaviour of the overall application.

Secondly, complex applications require a lot of planning and resources. Zope divides responsibilities into components which also allows *you* to divide up responsbilities when working in a team. Zope is an excellent platform for collaborative development.

Last but not least, the distinction between components with different responsibilities allows easy refactorings since only certain components have to be interchanged. A common example is the customization of presentation and layout components, meaning HTML pages and CSS style sheets, while the underlying software components are not touched.

securely publish objects on the web and other systems: Zope is about objects – it is object-oriented – and about the web. As the points before stated, one can use many Zope components in applications that are not web-specific. However, Zope's main intention is to allow people to manage objects that are published on the web as part of a larger application, not just a website.

Zope is about security, too. Compliance to international IT security norms is a major part of the Zope 3 philosophy. Zope 3 is therefore undergoing

a development-concurrent examination process run by an official IT security agency in Germany. As a result, the Common Criteria Certificate (ISO-15854), a security certificate accepted in Europe, North America and many other countries, will be issued to Zope as one of the first open source products. An invaluable by-product of this will also be that Zope's security model in its full extent is being documented in an ISO-compliant manner.

and make it easy to do Quality Assurance: The developers of Zope 3 have enforced a high level of quality during development. Zope itself is tested by several thousand automated tests; any modifications need to assure that the tests still pass, any new feature needs to be covered by new tests. This has not only lead to a great quality of the software itself, but makes Zope a great platform to do quality assurance with.

Zope runs on all major Unix platforms, including Mac OS X, as well as Microsoft Windows operating systems. It comes with its own webserver but can interoperate with an existing webserver software such as Apache.

Content Management

As an object-oriented web application server with features such as object persistency, Zope provides an excellent platform for building content management systems (CMS). Additional open source party libraries jointly developed by content management vendors allow one to easily construct a custom CMS built on Zope. It is important, though, to note that Zope itself is *not* a content management system. Yet, its whole machinery is geared toward managing content which is why the main focus of this book is the construction of applications that manage content.

Zope X3 vs. Zope 3

When we talk about Zope 3 as a software product, we are usually talking about *Zope X3*. When the development of Zope 3 was launched as a rewrite of Zope 2 from scratch, the goal was to first provide a software collection based on the new concepts and principles that led to the rewrite, *Zope X3*. Only later maybe a form of backward-compatability will be provided which should allow the porting of old Zope 2 packages. This will eventually lead to *Zope 3*. It is uncertain yet, when that will happen and what it will look like. As it was intended, Zope X3 is perfectly suitable for developing new applications, as demonstrated in this book. Packages that are developed with Zope X3 now will run on Zope 3 without modification.

1.3 The history of Zope

The Zope community might have been small when I joined it, but it certainly already had its myths. "Jim wrote the first Zope on a plane" the Zope birthing myth said. Fact is that Jim Fulton was not happy with the status-quo of web development in Python when he returned from the International Python Conference in 1998. Noone can blame him since he had just given a tutorial on CGI programming at the conference, certainly not the nicest way of web developing. Back at Digital Creations, he started writing Bobo, an object database and object request broker capable of publishing Python objects over the web. Later, the commercial Principia, Zope's predecessor, joined its little brother Bobo. Until today, method and attribute names like `bobobase_modification_time` or `isPrincipiaFolderish` in the Zope 2 API still tell that history of Zope's parents.

In 1998, Hadar Pedhazur of the venture capital firm Verticality Investment Group invested into Digital Creations and convinced them to make their successful web applications open source. Paul Everitt, CEO of Digital Creations at the time, announced the opening of the sources at the Python Conference 1998. That same year, Bobo and Principia became Zope, the *Z Object Publishing Environment*, which was released under the Zope Public License.

Zope's path of success began with version 2.0 which was released in 1999. It was the basis for all stable releases for around five years until Zope 3 was released. During that time, instead of selling Bobo and Principia, Digital Creations – later renamed to Zope Corporation – successfully sold its services as a software consultant and implemented solutions based on open source Zope, just like many other companies around the globe who started basing their services on Zope. Many successful open source Zope applications such as Plone, Silva, and CPS originally come from these companies and make up for a great part of the Zope community now. The wide-spread network of solution providers ensure every level of support that one can expect from an enterprise solution system.

1.4 The Python Programming Language

> *Zope magic is Python magic.*
>
> Jim Fulton at the EuroZope Conference Berlin 2002

Zope profits from Python's supremity. Even more, Zope would be impossible without Python. Python is a quick, easy-to-learn, Swiss Army knife-like scripting language – but so is Perl. Python also is an industrial strength, highly dynamic, object-oriented, interpreted programming language – but so is Java. What makes Python special then?

Python is easy to learn. It took me one afternoon to go through the Python tutorial [9]. Sure, I bought a few books on Python afterwards, but not for learning the language. I merely wanted to know what it was capable of.

Python is easy to read. Go and open an arbitrary python source code file in Zope 3. Chances are good that you will understand most of what is going on without actually having seen Python before. Using indentation for marking blocks certainly is the most aesthetic aspect of the language, but also the one many people have to get used to first. Obviously, it is still possible to write cryptic code in Python, but less likely.

Python is easy to write. I rarely need the library reference when programming. When developing Java, I constantly find myself looking up standard library interfaces, abstract classes, etc. Python is not as wordy as Java, but it is not as cryptic as Perl, either.

Python is easy to develop with. Not having complicated package names and large names in standard library classes, Python can easily be developed without an IDE that would ease the development process. In fact, the Zope contributor community is more or less equally split into those who use emacs and those who use vi. Moreover, Python does not require a compiler. That abandons long compiling sessions.

Python comes with batteries included. The regular Python distribution comes with a rich library of useful modules, ranging from parsers for several text file formats to email messaging and processing. That means you can get started with Python very quickly without having to collect the basic modules yourself. In case you are looking for third party packages you can use the *Python Package Index*[1] to find the software package of your desire. There are numerous third party packages available as open source which can easily be used in Zope applications, too.[2]

"Python – it fits your brain" a popular Python t-shirt reads; it sums up my programming experience with Python in one sentence. If you are new to it, you are sure to be converted to Python by the end of this book. In case you are looking for a Python book to accompany your Zope 3 development, the author can strongly recommend the excellent *Dive Into Python* by Mark Pilgrim [3].

[1] Python Package Index <http://www.python.org/pypi>

[2] In Chapter 12, for example, we will use *ReportLab*, a third party library for PDF generation, to generate PDF documents out of Zope content objects.

1.5 About the examples

Whenever possible, the examples in this book follow a common theme, an example application. This example application is to drive the fictitious *World Cookery* website, a website where hobby cooks from around the globe can share their recipes with each other. This particular application was chosen because it incorporates the most important characteristics of the majority of Zope-based web applications:

- There is a limited set of content object types. In the case of our example application, the primary type of content object is a *recipe*.
- Content is added in a management interface accessible through-the-web using a web browser.
- The layout of the application follows a common theme, for example a *corporate identity*.
- The application has to cope with multiple or even numerous users.
- The target audience is international, thus internationalization is required.
- Existing features must easily be extensible and new functionality easily be addable.

Because the example application is extended throughout the book, there is a version of it available for each chapter. To work with the example code of a particular chapter, simply copy the corresponding directory from the examples archive to a directory in your Python modules search path, for example your instance's `lib/python` directory, and rename it to `worldcookery` so that the package is found under the right name. See Chapter 2 for more information on installing Zope and add-on packages.

You may download an archive containing the example application from the book's website[3].

Interactive interpreter examples

As an interpreted language, Python provides us with an interactive interpreter, which is invoked by running the `python` program without arguments. This interpreter shell allows us to test code or conduct quick experiments on it. Thus you will find examples of code being typed in at the Python interpreter prompt. These lines start with `>>>`. If you choose not to type the sometimes admittedly lengthy example listings into your computer, but to download them, consider at least typing in these interactive interpreter sessions. It will give you a better understanding of what is going on.

You can invoke the Python interpreter usually by typing `python` at the command line. On Windows, you often need to explicitly call it from the directory it is installed in; that means you can either type something similar to `C:\Python23\python` or put the `C:\Python23` directory in the command line search path (`PATH` environment variable).

[3] Book website <http://www.philikon.de/book>

Also make sure that Zope's Python packages are in the interpreter's search path. This should not be a problem for Windows users, only on a Unix system you will probably have to set the `PYTHONPATH` environment variable, depending on where you installed the Zope software files. To test whether the interpreter can find the Zope libraries, invoke the interpreter and try to import the `zope` package:

```
philipp@bender:~$ python
Python 2.3.4 (#1, Jun 12 2004, 15:25:34)
[GCC 3.3 20030304 (Apple Computer, Inc. build 1640)] on
darwin
Type "help", "copyright", "credits" or "license" for
more information.
>>> import zope
Traceback (most recent call last):
...
ImportError: No module named zope
```

If you see this error message, then you need to set the search path variable. Quit the interpreter by hitting **Ctrl**–**D** and enter the following command:

```
~$ export PYTHONPATH=/usr/local/ZopeX3-3.x.y/lib/python
```

Now try again invoking the Python interpreter and importing the `zope` package.

Some interactive interpreter examples in this book require you having started the interpreter from your Zope instance directory. See Chapter 2 for more information on instances.

Conventions

When you encounter a particular example listing, the caption of the listing states the name of the file whose contents in shown in parentheses. For example, if an example is titled *Configuring a content type (`configure.zcml`)*, then look for the code in the `configure.zcml` file within that chapter's examples directory. If you installed the chapter's examples directory as the `worldcookery` package in the Zope instance, the path relative to the instance directory will be `lib/python/worldcookery/configure.zcml`.

The coding style and naming conventions of the Python code follow the Zope 3 coding style and naming conventions [2], Guido van Rossum's recommendations [8], and the author's personal taste (in that order).

Features in future releases

Zope is under constant development. An active community of developers constantly enhance and improve its. That means that features that are used and documented now might become obsolete in upcoming versions because they are replaced with other, better features. Alone by the time of this writing, a hand full of features were deprecated or about to be deprecated for future versions.

Why am I documenting them here? Because Zope X3.0, the Zope 3 version that this book is based on, can be used *now*. Unlike the development code, X3.0 is being used in production successfully. Even though a couple of features have been deprecated, they are known to work well and satisfy existing needs. Constant improvement is a good thing. It shows that Zope development is not standing still. Having stable versions to work with is important, too, though. The Zope 3 development cycle guarantees us both.

The following features, some well known to Zope 2 users, are still being developed on and are very likely to appear in upcoming versions of Zope 3 and if possible in upcoming editions of this book:

Indexing and searching The successor to the famous Zope Catalog and the necessary indices were still being developed at the time of this writing.

Sessions A thread-safe session framework has been added to Zope 3 already and will be released with the next release, most likely X3.1.

Relational database access Zope X3.0 already supports some functionality that allows access to relational databases. Most functionality that makes RDB integration a lot easier is not in the Zope core, though, and thus out of this book's scope.

Workflow A rudimentary workflow implementation already exists in the development tree of Zope 3. It is probably not usable for production systems, though. Different concepts of modelling workflow are being discussed and implemented for trial which might eventually lead to a usable workflow framework.

XML Even though Zope 3 itself uses XML in quite a few places (Page Templates, ZCML, etc.), it does not yet have a framework dedicated to XML processing. There have been initial prototypes, however, which were by the time of this writing being developed in external projects.

It's Open Source!

If you think that a vital feature is missing from Zope, do not hesistate to participate in development. Everyone can contribute and the community welcomes even the smallest contribution.